NXP Semiconductors
Reference Manual

Document identifier: MWCT2xxxS_RM
Rev. 3, 09/2024

MWCT2xxxS Reference Manual

Supports MWCT2014S,MWCT2015S, MWCT2016S, MWCT2D16S, and MWCT2D17S

Contents

Chapter 1 About This Manual..... 7

Chapter 2 Introduction..... 15

Chapter 3 Memory Map..... 29

Chapter 4 Signal Multiplexing..... 33

Chapter 5 Cortex-M7 Overview..... 45

Chapter 6 Miscellaneous Control Module (MCM)..... 49

Chapter 7 Miscellaneous System Control Module (MSCM).....68

Chapter 8 Virtualization Wrapper (VIRT_WRAPPER)..... 106

Chapter 9 System Integration Unit Lite2 (SIUL2).....125

Chapter 10 Touch Sensing Pin Coupling (TSPC).....261

Chapter 11 Crossbar Switch (AXBS)..... 268

Chapter 12 Peripheral Bridge (AIPS_Lite)..... 281

Chapter 13 Direct Memory Access Multiplexer (DMAMUX)..... 283

Chapter 14 Enhanced Direct Memory Access (eDMA).....293

Chapter 15 Interrupt Monitor (INTM).....396

Chapter 16 Semaphores2 (SEMA42)..... 405

Chapter 17 Crossbar Integrity Checker (XBIC).....414

Chapter 18 Extended Resource Domain Controller (XRDC)..... 426

Chapter 19 Memory and Memory Interfaces..... 501

Chapter 20 Embedded Flash Memory (c40asf)..... 507

Chapter 21 Flash Memory Controller (PFLASH)..... 569

Chapter 22 RAM Controller (PRAMC)..... 595

Chapter 23 Clocking..... 601

Chapter 24 Clock Generation Module (MC_CGM)..... 669

Chapter 25 Fast Internal RC Oscillator (FIRC)..... 761

Chapter 26 Slow Internal RC Oscillator (SIRC)..... 764

Chapter 27 Fast Crystal Oscillator Digital Controller (FXOSC).....767

Chapter 28 Slow Crystal Oscillator Digital Controller (SXOSC).....774

Chapter 29 PLL Digital Interface (PLLDIG).....778

Chapter 30 Reset Overview.....793

Chapter 31 Boot Overview.....817

Chapter 32 Reset Generation Module (MC_RGM)..... 837

Chapter 33 Power-on Reset Watchdog (POR_WDG)..... 867

Chapter 34 Security Overview..... 870

Chapter 35 Hardware Security Engine (HSE_B)..... 872

Chapter 36 Device Configuration Format (DCF) records.....892

Chapter 37 Device Configuration Module General-Purpose Registers (DCM_GPR)..... 897

Chapter 38 Device Configuration Module (DCM)..... 1003

Chapter 39 Messaging Unit (MU)..... 1049

Chapter 40 Power Management..... 1186

Chapter 41 Power Management Controller (PMC for MWCT2D17S and MWCT2D16S)..... 1205

Chapter 42 Power Management Controller (PMC for MWCT2016S)..... 1217

Chapter 43 Mode Entry Module (MC_ME)..... 1230

Chapter 44 Power Control Unit (MC_PCU)..... 1339

Chapter 45 Wakeup Unit (WKPU)..... 1342

Chapter 46 Safety Overview..... 1367

Chapter 47 Error Injection Module (EIM)..... 1377

Chapter 48 Error Reporting Module (ERM)..... 1433

Chapter 49 Fault Collection and Control Unit (FCCU)..... 1466

Chapter 50 Self-test General-Purpose Registers (SELFTEST_GPR)..... 1524

Chapter 51 Self-Test Control Unit (STCU2)..... 1528

Chapter 52 Register Protection (REG_PROT)..... 1573

Chapter 53 Clock Monitoring Unit – Frequency Check (CMU_FC)..... 1586

Chapter 54 Clock Monitoring Unit – Frequency Meter (CMU_FM)..... 1601

Chapter 55 Cyclic Redundancy Check (CRC)..... 1609

Chapter 56 Power Conversion and Motor Control (PCMC)..... 1620

Chapter 57 Analog-to-Digital Converter (ADC)....1637

Chapter 58 Low Power Comparator (LPCMP)....1798

Chapter 59 Logic Control Unit (LCU)....1838

Chapter 60 Enhanced Modular IO Subsystem (eMIOS)....1914

Chapter 61 Body Cross-triggering Unit (BCTU)....2001

Chapter 62 Trigger MUX (TRGMUX)....2039

Chapter 63 Software Watchdog Timer (SWT)....2101

Chapter 64 System Timer Module (STM)....2117

Chapter 65 Periodic Interrupt Timer (PIT)....2126

Chapter 66 Real Time Clock (RTC)....2147

Chapter 67 Low Power Serial Peripheral Interface (LPSPI)....2160

Chapter 68 Low Power Inter-Integrated Circuit (LPI2C)....2211

Chapter 69 Flexible I/O (FlexIO)....2270

Chapter 70 CAN (FlexCAN)....2346

Chapter 71 Synchronous Audio Interface (SAI)....2520

Chapter 72 Ethernet Media Access Controller (EMAC)....2566

Chapter 73 Low Power Universal Asynchronous Receiver/Transmitter (LPUART)....3155

Chapter 74 Quad Serial Peripheral Interface (QuadSPI)....3203

Chapter 75 Debug Subsystem....3288

Chapter 76 JTAG Controller (JTAGC)..... 3332

Chapter 77 JTAG Data Communication (JDC)..... 3341

Chapter 78 Temperature Sensor (TempSense)..... 3350

Appendix A Revision History.....3357

Legal information..... 3364

Chapter 1
About This Manual

1.1 Audience

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

1.2 Organization

This manual has two main sets of chapters.

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

1.2.1 Attachments

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

NOTE

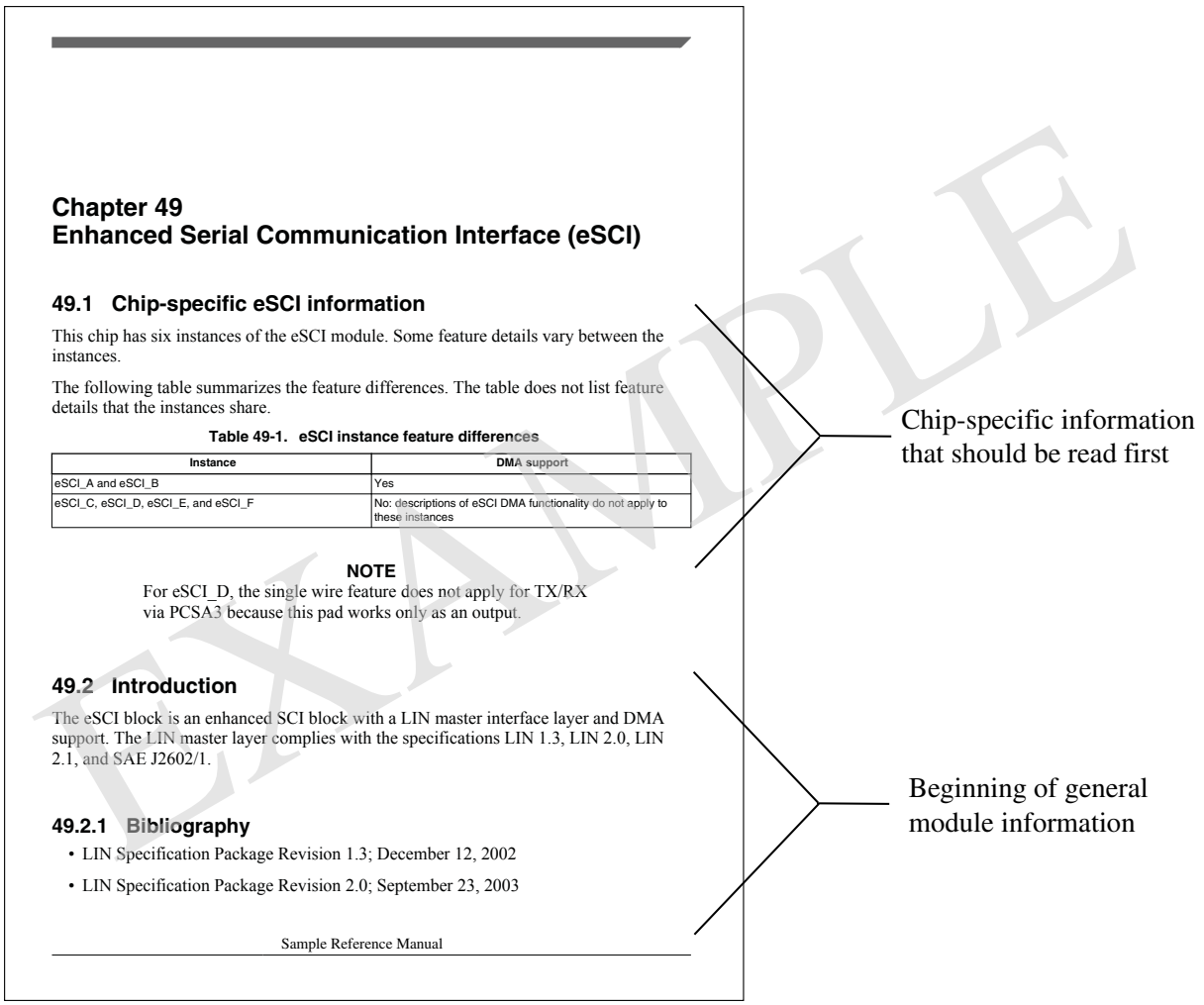
Select the paperclip icon on the left side of the PDF window to see the list of attachments.

1.3 Module descriptions

Each module chapter has two main parts:

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

The following figure shows you an example of this demarcation.



Chapter 49
Enhanced Serial Communication Interface (eSCI)

49.1 Chip-specific eSCI information

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

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

Table 49-1. eSCI instance feature differences

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

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

49.2 Introduction

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

49.2.1 Bibliography

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

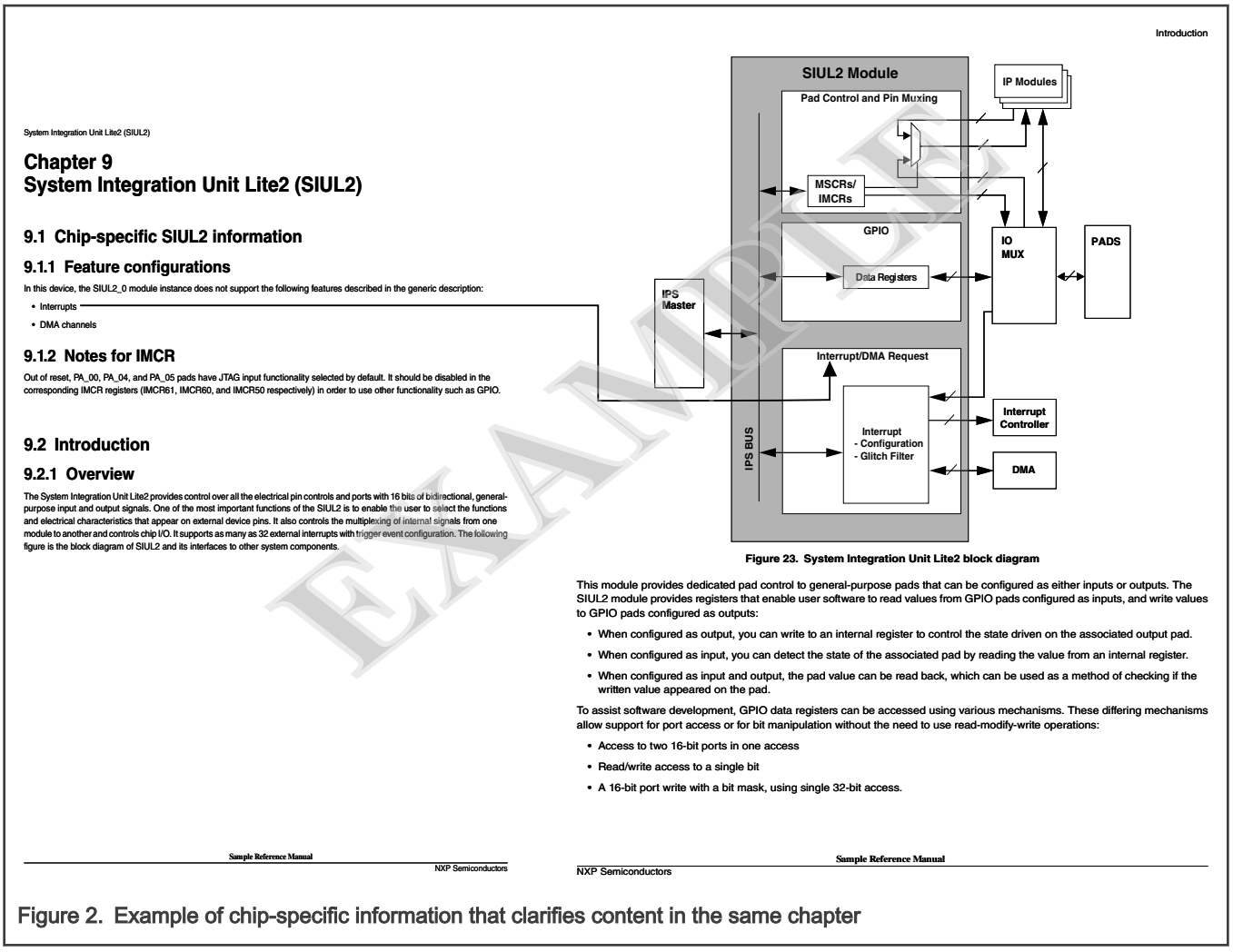
Sample Reference Manual

EXAMPLE

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

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

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



Introduction

System Integration Unit Lite2 (SIUL2)

Chapter 9
System Integration Unit Lite2 (SIUL2)

9.1 Chip-specific SIUL2 information

9.1.1 Feature configurations

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

- Interrupts
- DMA channels

9.1.2 Notes for IMCR

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

9.2 Introduction

9.2.1 Overview

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

Figure 23. System Integration Unit Lite2 block diagram

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

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

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

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

Sample Reference Manual
NXP Semiconductors

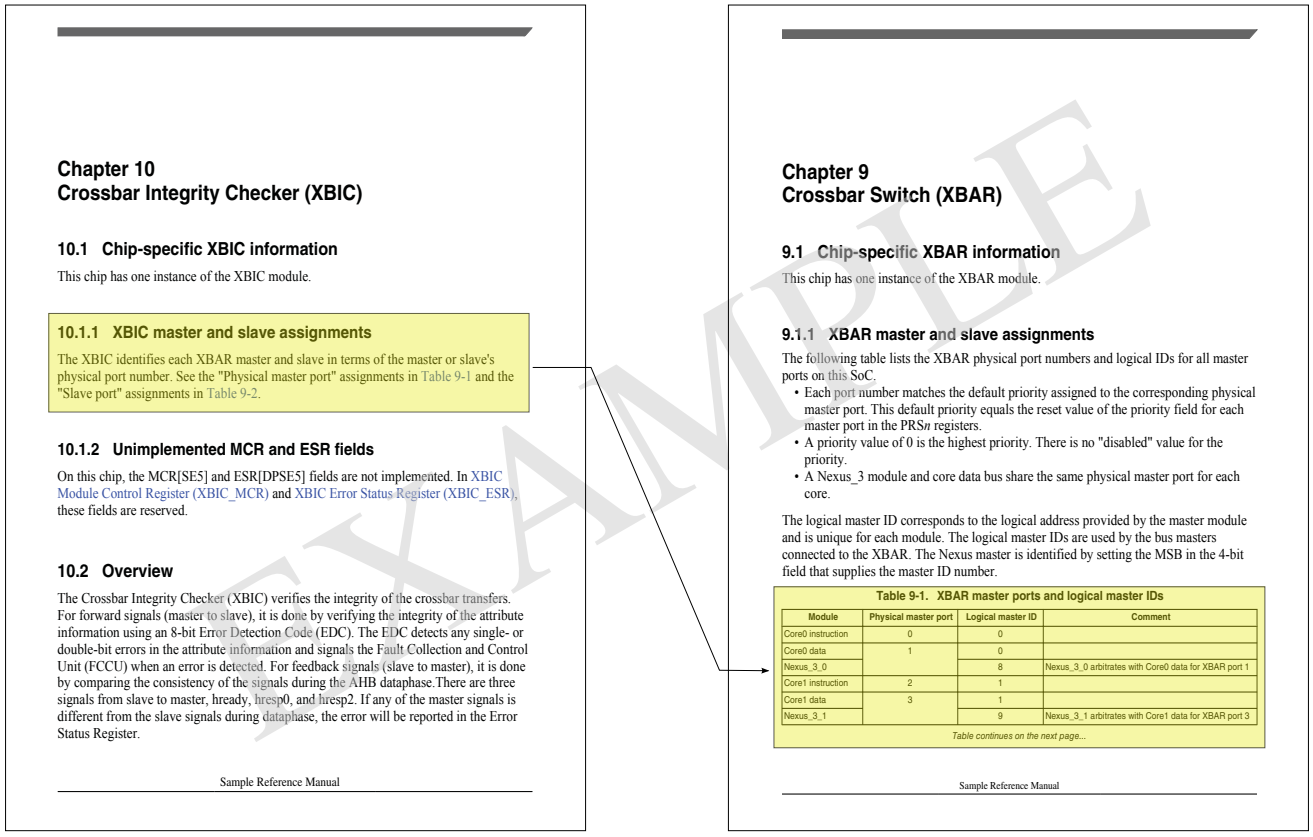
Sample Reference Manual
NXP Semiconductors

EXAMPLE

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

1.3.2 Chip-specific information that refers to a different chapter

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



Chapter 10
Crossbar Integrity Checker (XBIC)

10.1 Chip-specific XBIC information

This chip has one instance of the XBIC module.

10.1.1 XBIC master and slave assignments

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

10.1.2 Unimplemented MCR and ESR fields

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

10.2 Overview

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

Sample Reference Manual

Chapter 9
Crossbar Switch (XBAR)

9.1 Chip-specific XBAR information

This chip has one instance of the XBAR module.

9.1.1 XBAR master and slave assignments

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

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

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

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

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

Table continues on the next page...

Sample Reference Manual

EXAMPLE

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

1.4 Register descriptions

Module chapters present register information in the following:

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

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

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

Figure 4. Register figure conventions

NOTE

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

1.5 Conventions

1.5.1 Notes and cautions

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

NOTE

Emphasizes information that deserves extra attention.

CAUTION

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

1.5.2 Numbering systems

The following suffixes identify different numbering systems:

Table 1. Numbering systems

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

1.5.3 Typographic notation

The following typographic notations are used throughout this document:

Table 2. Typographic notation

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

Table continues on the next page...

Table 2. Typographic notation (continued)

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

1.5.4 Special terms

The following terms have special meanings.

Table 3. Special terms

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

1.6 Editorial changes

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

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

1.7 Glossary

The "Glossary" section provides a list of selected terms and definitions used in a chapter. The glossary, if present, is placed at the end of the chapter.

A glossary contains:

- NXP-specific terms, symbols, and notations, with definitions.
- Terms, symbols, and notations that can have different meanings in different contexts. The glossary defines the way they are used in the chapter—especially if the meaning is different from the meaning in standard references.

A glossary does not contain:

- Field mnemonics
- Register mnemonics
- Module mnemonics
- Signal names
- Industry-standard terms

Chapter 2
Introduction

2.1 Overview

The MWCT2xxS product series further extends the highly-scalable portfolio of Arm ® Cortex ® - M0+/M4F MWCT1xxx chips in the automotive industry with the Arm Cortex-M7 core at higher frequency, more memory, ASIL-B rating and advanced security module. With a focus on automotive environment robustness, the MWCT2xxS product series devices are well suited to a wide range of applications in electrical harsh environments, and are optimized for cost-sensitive applications offering new, space saving package options. The MWCT2xxS series offers a broad range of memory, peripherals and performance options. Devices in this series share common peripherals and pin-out, allowing developers to migrate easily within a chip series or among other chip series to take advantage of more memory or feature integration.

2.2 MWCT2xxxS product series

The MWCT2xxxS series comes to market together with easy-to-use enablement software, application specific software, and various development tools, supported by broad third parties in different development phases.

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

Following are the general features of the MWCT series chips:

- 32-bit Arm Cortex-M7 core with IEEE-754 compliant SPFPU, executing up to 160 MHz
- Scalable memory footprints up to 4 MB flash memory and up to 512 KB SRAM
- Precision mixed-signal capability with low power comparators (LPCMP) and multiple 12-bit ADCs
- Powerful timers for a broad range of applications including wireless charging, lighting control and body applications
- Serial communication interfaces such as Serial communication interfaces including LPUART, LPSPI, LPI2C, FlexCAN with ISOCAN-FD support, Ethernet and QuadSPI. FXIO configuration allows other communication options including SENT.
- EVITA full and light functionality compliant HSE_B
- Power supply (3.0 – 5.5 V) with fully functional flash memory program/erase/read operations
- Functional safety compliance with ISO26262 levels B (features depend on device specification)
  - Lockstep cores
  - Multiple internal watchdogs
  - Voltage monitors
  - Clock monitors
  - Memory protection
  - Data transport checks
  - ECC on memories
  - Cyclic redundancy checking
- Ambient operation temperature range: –40°C to 125°C
- Junction temperature range: –40°C to 150°C
- Tools solutions:
  - MWCT Design Studio IDE
  - NXP GCC compilers for Cortex-M7 and IDE plugins for partner compilers support

- Low cost debugger with multi-core debug, tracing and profiling support
- S32 Configuration Tools:
  - Pin wizard with detailed pin configuration report
  - Graphical clock tree for clock configuration
  - Peripheral configuration
- Model-Based Design Toolbox:
  - Integrated Simulink®-embedded target for direct rapid prototyping and PIL development workflows
  - Peripheral device interface blocks and drivers
  - Target-optimized math and motor control algorithm blocks for efficient execution on the target chip
  - Bit-accurate simulation results in the Simulink® simulation environment
- FreeMaster

- Software solutions:
  - RTD (Real Time Drivers)
    - Autosar 4.4 compliant MCAL plus complex device drivers for external ICs (e.g. PMIC) and miscellaneous SW components (e.g. inter core communication)
    - Low level drivers covering all device periphery
    - Example projects for each driver
    - Configuration components for Tresos Studio and S32ConfigTools framework
    - Integrated with MWCT Design Studio
  - Stacks/libraries
    - Communication stacks:
      - LIN, TCP/IP, gPTP, AVB, WiFi, AWS IOT
    - Security:
      - mBedTLS
    - Safety:
      - Safety Software Framework
    - Real time control:
      - AMMCLib
      - ISELED
    - RTOS:
      - FreeRTOS
    - Firmware:
      - Security firmware for on-chip crypto engine
    - Core to core communication:
      - IPFC
  - Designed to work in conjunction with NXP SBC UJA124x and SBC FS2600

2.3 Feature summary

The MWCT2xxxS product family includes the Arm Cortex-M7 core features described in the following table.

Table 4. MWCT2xxxS chip's feature summary

Feature	Inclusions
Core and architecture	• Arm Cortex-M7 core running up to 160 MHz • Arm core based on the Armv7 architecture and ThumbR-2 ISA with 2.14 DMIPS/MHz • Upto 16 KB data and 16 KB instruction cache for optimizing wait state execution from memories • 96 KB Tightly Coupled Memory associated with each core • On-core MPU for dynamic task protection (16 regions) • SPFPU, IEEE 754 compliant • Harvard bus architecture implementing dedicated instruction and data path • 5-stage pipeline with branch speculation • XRDC integrated with a crossbar switch to provide memory and peripheral protection • DSP • I/O protection (VIRT_WRAPPER) • ETM supporting instruction trace • Arm third-party ecosystem support: software and tools to help minimize development time and cost
DMA	• Up to 2x64-channel DMAMUX • eDMA with up to 32 channels • Complex data transfers performed with minimal intervention from a host processor • Programmable support for scatter-gather DMA processing
System and power management	• Support for simplified power modes (Run and Standby) • Support for clock gating of unused modules; specific peripherals continue to work in low-power modes • Support for external ballast transistor to generate core supply • Fully independent CPU and peripheral clocking scheme • Rapid start-up from a 48 MHz FIRC • Various low-power oscillators such as the 32 kHz SIRC and an external 32 kHz crystal support SXOSC • PMC with LVD and selectable trip points • Support for multiple power modes • NMI
Memory and memory interfaces	• Up to 4 MB program flash memory, up to 128 KB data flash memory, and up to 512 KB SRAM, all with an ECC

Table continues on the next page...

Table 4. MWCT2xxxS chip's feature summary (continued)

Feature	Inclusions
	• 4-bit/8-bit QuadSPI
Clocks	• External 8 MHz–40 MHz crystal oscillator or resonator • External 32 kHz crystal oscillator • Internal clock references — 48 MHz FIRC ±5% — 32 kHz SIRC ±10% • Up to 960 MHz PLL for divided system clock operation
Security and integrity	• Hardware security engine (HSE_B) — Upgradable Firmware delivered by NXP, to be programmed by the user • Security ciphers: — Symmetric: AES-128/192/256 — Cipher modes: ECB, CBC, CMAC, GMAC, CTR, OFB, CCM, and GCM — Asymmetric: RSA (up to 4096 bits) and ECC (up to 521 bits) — Hash: Miyaguchi-Preneel, SHA-2/SHA-3 (up to 512 bits) — Number of keys is configurable and controlled by HSE FW — Random number generator • Security use case supported: — Secure boot — Secure communication — Component protection — Secure storage — Key exchange
Safety ISO26262	• Classification up to ASIL-D • ERM and EIM support • WDOG with an independent clock source • Voltage monitors • Bandgap voltage available as ADC input • External clock source monitoring using an independent reference • PLL lock and loss-of-lock protection • XRDC • ECC on code flash memory, data flash memory, and system RAM

Table continues on the next page...

Table 4. MWCT2xxxS chip's feature summary (continued)

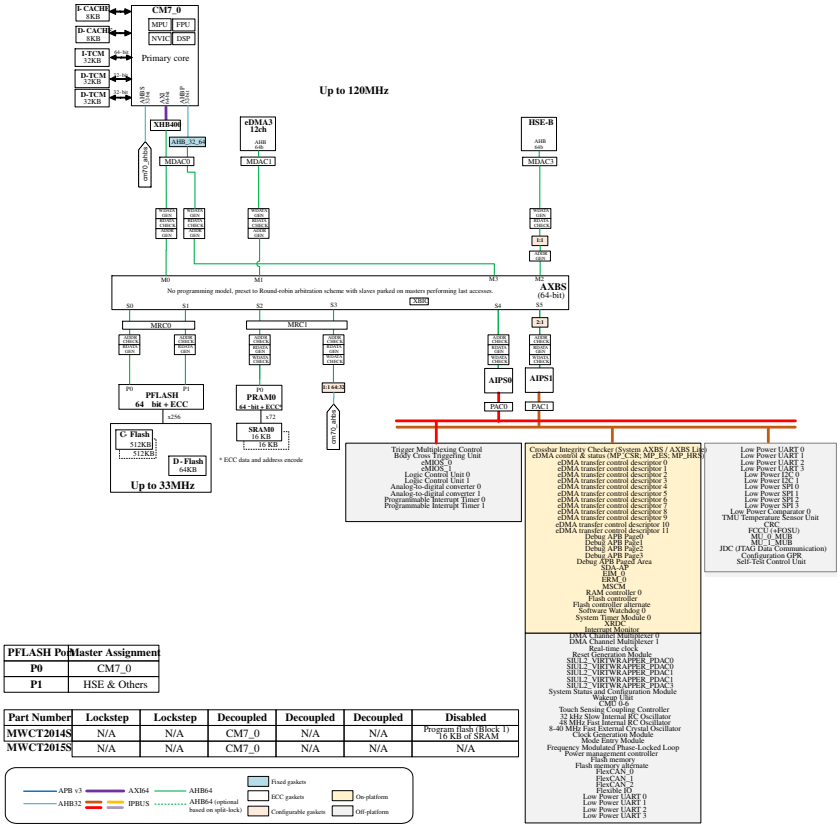
Feature	Inclusions
	• ADC self-test feature • Internal analog monitoring of all supplies available • CRC generation module • FCCU failure output
Analog	• 12-bit ADC — Up to 72 external analog inputs — 1 µs conversion time — Internal bandgap voltage reference channel, supporting automatic compare and an optional hardware trigger — Up to five internal reference inputs — Automatic compare with interrupt — Self-test and self-calibration scheme • ACMP with an internal 8-bit DAC as a reference — ACMP with both positive and negative inputs, separately selectable interrupts on rising and falling comparator output — Ability to cross trigger the timers from both the ADC and ACMP outputs • Temperature Sensor (TempSense) with output measurable by ADC.
Timers	• 16-bit or 24-bit eMIOS timers, offering up to 72 standard channels — Input capture, output compare, and PWM modes — Fault input support with global fault control — Multiple features such as deadtime insertion, configurable polarity, quadrature decoding, and so on • 32-bit PITs, with four channels, for raising interrupts and triggering DMA channels • 32-bit RTC • Motor control and power conversion using combination of eMIOS, LCU , BCTU, and ADC • Up to four STMs, with four channels each
Communications	• LPSPI supporting DMA with full-duplex or single-wire bidirectional communication in Master or Slave mode • FlexIO, with an option to configure it as different communication peripherals, offering support for SENT • LPI2C modules with DMA support, low-power availability, master or slave support, and system management bus

Table continues on the next page...

Table 4. MWCT2xxxS chip's feature summary (continued)

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

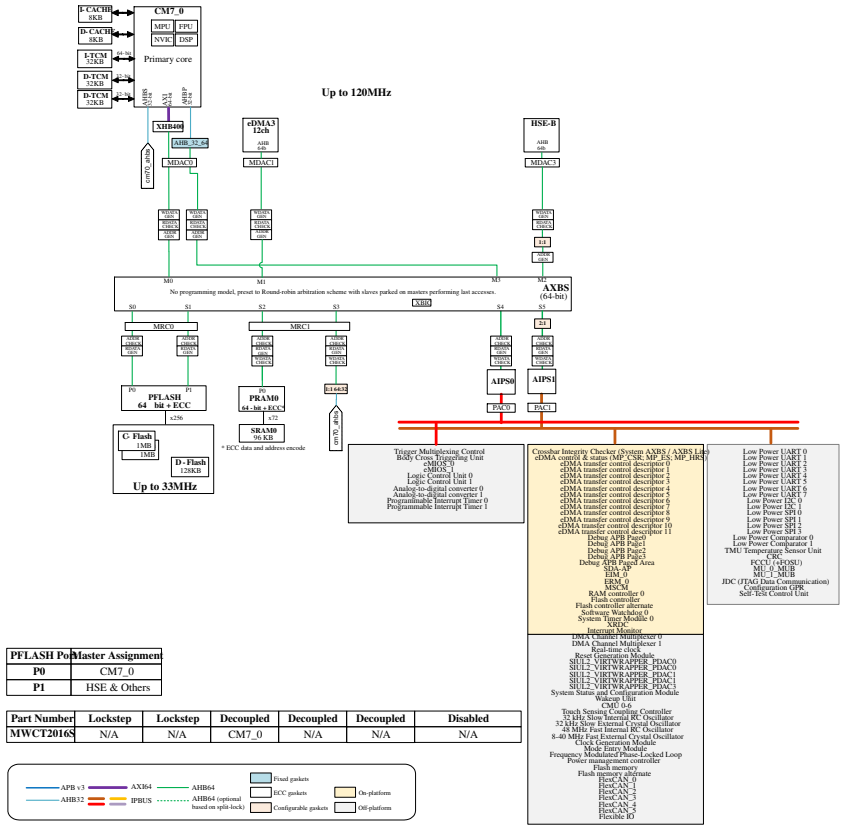
2.4 Block diagram



PFLASH Port	Master Assignment
P0	CM7_0
P1	HSE & Others

Part Number	Lockstep	Lockstep	Decoupled	Decoupled	Decoupled	Disabled
MWCT2014S	N/A	N/A	CM7_0	N/A	N/A	Program flash (Block 1) 16 KB of SRAM
MWCT2015S	N/A	N/A	CM7_0	N/A	N/A	N/A

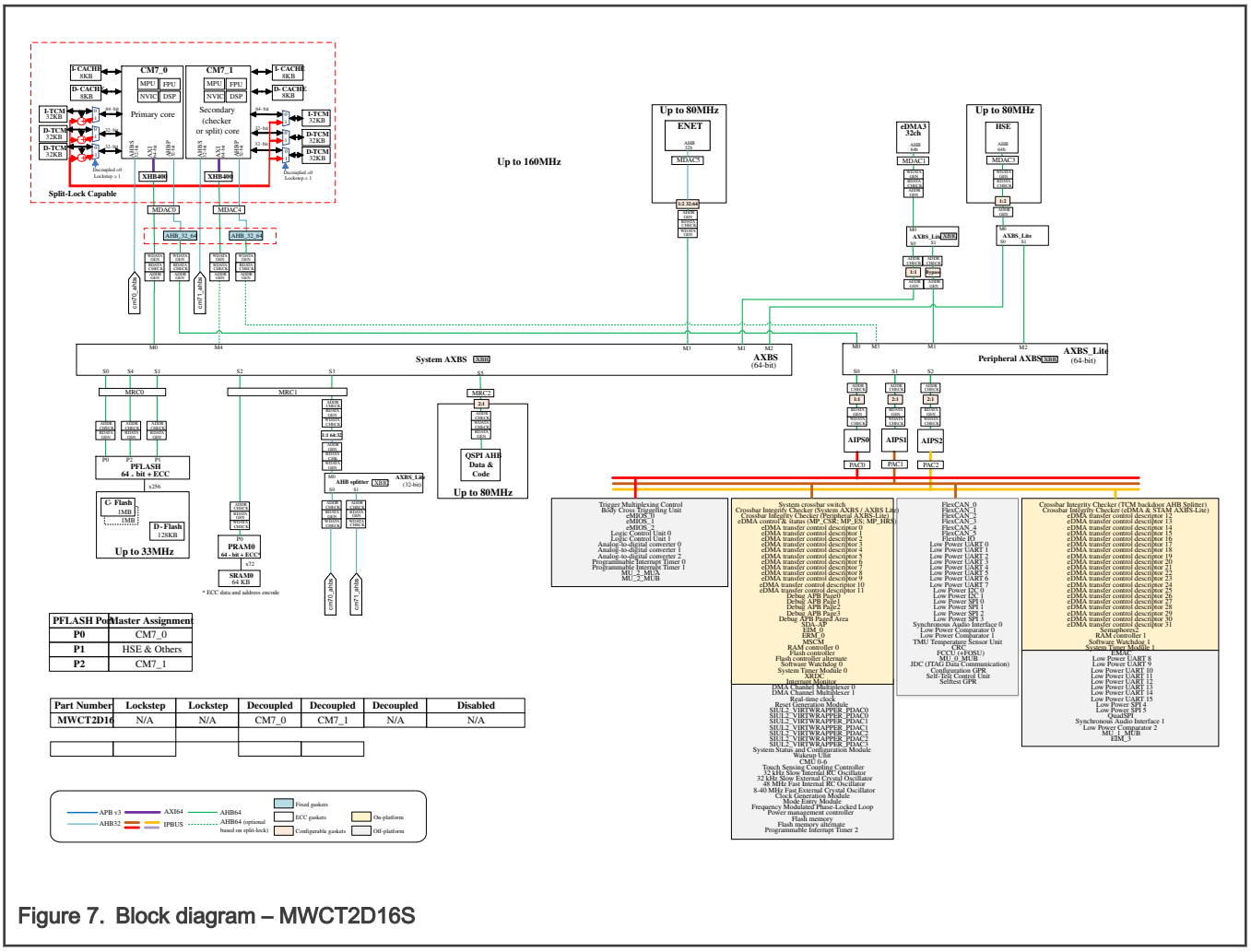
Figure 5. Block diagram – MWCT2015S and MWCT2014S



PFLASH Port	Master Assignment
P0	CM7_0
P1	HSE & Others

Part Number	Lockstep	Lockstep	Decoupled	Decoupled	Decoupled	Disabled
MWCT2016S	N/A	N/A	CM7_0	N/A	N/A	N/A

Figure 6. Block diagram – MWCT2016S



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

Part Number	Lockstep	Lockstep	Decoupled	Decoupled	Decoupled	Disabled
MWCT2D16	N/A	N/A	CM7_0	CM7_1	N/A	N/A

Figure 7. Block diagram – MWCT2D16S

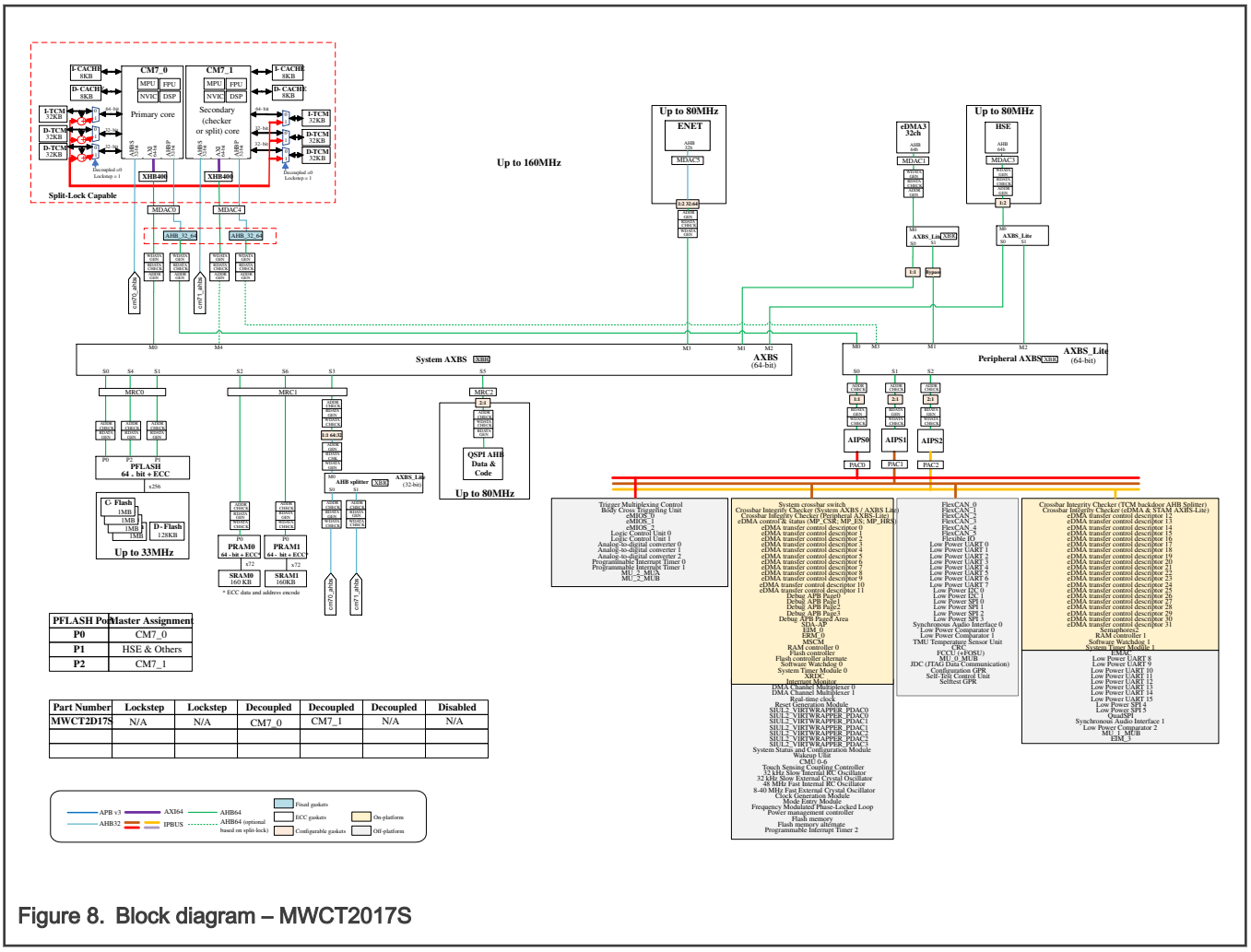


Figure 8. Block diagram – MWCT2017S

2.5 Feature comparison

The following table compares some of the prominent features related to memory and package options of these chips from the MWCT2xxxS family/product series:

- MWCT2014S
- MWCT2015S
- MWCT2016S/MWCT2016SG
- MWCT2D16S
- MWCT2D17S

Table 5. MWCT2xxxS chip's feature comparison

Feature	Chip				
	MWCT2014S	MWCT2015S	MWCT2016S/ MWCT2016SG	MWCT2D16S	MWCT2D17S
Safety/ASIL	B	B			
Program flash memory	512 KB	1 MB	2 MB		4 MB
Data flash memory (KB)	64	64	128		
Total RAM (KB)	112KB (incl. 96KB TCM)	128KB (incl. 96KB TCM)	192KB (incl. 96KB TCM)	256KB (incl. 192KB TCM)	512KB (incl. 192KB TCM)
Standby RAM	32KB				
Security	HSE_B				
Core quantity	1 x M7			2 x M7	2 x M7
Frequency (MHz)	120			160	
DMA channels	12			32	
ASIL–B DMIPS[1]	277–387			739–1033	739–1033
ASIL–D DMIPS[1]	—	—			
ASIL–B CoreMark score[2]	634			1692	1692
ASIL–D CoreMark score[2]	—	—			
FlexCAN instances	3		6	4	6
EMAC instances	—	0		1	
GMAC instances	—	0			
SAI instances	—	0		2	
LPUART instances	4		8	4	16
LPSPI instances	4				6
I^2C instances	2				
FlexIO (incl. SENT support) channels	16		32		

Table continues on the next page...

Table 5. MWCT2xxxS chip's feature comparison (continued)

Feature	Chip				
	MWCT2014S	MWCT2015S	MWCT2016S/ MWCT2016SG	MWCT2D16S	MWCT2D17S
QuadSPI instances	—			1[3]	
uSDHC instances	—				
ADC instances	2				3
LPCMP instances	1		2		3
PIT instances	2			3	
SWT instances	1			2	2
STM instances	1			2	
LCU instances	2				
BCTU instances	1				
TRGMUX instances	1				
eMIOS instances	2				3
RTC instances	1				
172-HDQFP package[4]	No		Yes		
100-HDQFP package	Yes				No
48-pin LQFP package	Yes		No		

1. Final DMIPS is in range based on compiler setting. Low number is using "ground rules" laid out in the Dhrystone documentation. High number is using inlining of functions.
2. Result depends on specific compiler version, contact NXP sales representative for more details.
3. 4-bit data width
4. 172-HDQFP-EP (exposed pad)

2.6 Glossary

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

uSDHC	Ultra Secure Digital Host Controller
WDOG	Windowed watchdog

Chapter 3
Memory Map

3.1 Introduction

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

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

3.2 SRAM memory map

The memory map file attached to this document provides a complete architectural address space definition for various sections that the RAM is partitioned into and across the MWCT2xxxS product series. Based on the physical sizes of the memories and peripherals, the actual address regions used may be smaller. For details see chapter 'Memory and Memory Interfaces'.

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

The Cortex-M7 core can access these memories sequentially:

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

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

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

3.4 TCM as system memory

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

1. Write 1 to MC_ME's PRTN2_COFB1_CLKEN[REQ62] field for Cortex-M7_0, PRTN2_COFB1_CLKEN[REQ63] field for Cortex-M7_1, PRTN2_COFB2_CLKEN[REQ64] field for Cortex-M7_2, and PRTN2_COFB2_CLKEN[REQ65] field for Cortex-M7_3. This enables the Cortex-M7 core's TCM

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

Table 6. TCM modes of operation

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

Table continues on the next page...

Table 6. TCM modes of operation (continued)

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

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

3.5 Considerations related to TCM's implementation

You must first initialize TCM (ITCM and DTCM) and system RAMs by 64-bit writes before performing read accesses. The system RAM can be initialized using eDMA and core. The ITCM initialization can be performed only by core using either direct or back-door accesses. The DTCM can be initialized also by 32-bit writes performed either using core's direct and back-door accesses as or eDMA. These writes are required to set up the initial ECC code words after chip power-on reset.

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

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

3.6 Flash memory map

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

3.7 AIPS-Lite memory map

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

Table 7. Regions associated with peripheral space

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

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

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

3.8 Serialization of memory operations

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

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

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

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

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

3.9 PPB memory map

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

Table 9. PPB memory map

Starting hex address	Ending hex address	Size (KB)	Module
E000_0000	E000_0FFF	4	ITM
E000_1000	E000_1FFF		DWT
E000_2000	E000_2FFF		FPB
E000_3000	E000_DFFF	44	—
E000_E000	E000_EFFF	4	SCS
E000_F000	E003_FFFF	196	Reserved
E004_0000	E004_0FFF	4	TPIU
E004_1000	E004_1FFF		ETM
E004_2000	E004_2FFF		CTI
E004_3000	E004_3FFF		—
E004_4000	E004_4FFF		
E004_5000	E004_5FFF		
E004_6000	E007_FFFF	232	
E008_0000	E008_0FFF	4	MCM
E008_1000	E008_1FFF		—
E008_2000	E008_2FFF		
E008_3000	E00F_EFFF	496	
E00F_F000	E00F_FFFF	4	Cortex-M7 PPB ROM table

3.10 Glossary

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

Chapter 4
Signal Multiplexing

4.1 Introduction

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

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

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

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

NOTE

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

4.2 Pad description

Following figure shows the basic representation of a GPIO pad.

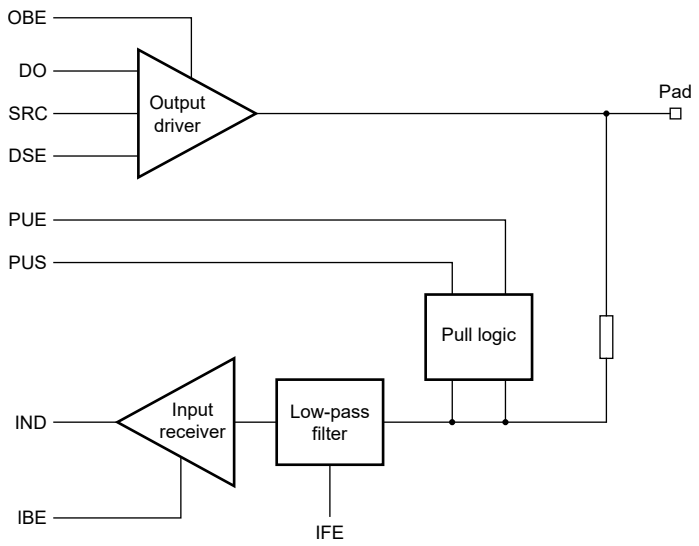


Figure 9. GPIO pad representation

Table 10. Pad Signal description

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

Table continues on the next page...

Table 10. Pad Signal description (continued)

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

Table 11. Input buffer enable

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

Table 12. Output buffer enable

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

Table 13. Input filter enable

IFE	Description
1	Input filter enabled
0	Input filter disabled

Table 14. Pull up/Pull down

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

Table 15. Slew rate control

SRC	Description
0	Slew rate enabled
1	Slew rate disabled

Table 16. Drive strength enable

DSE	Description
1	Drive strength supported
0	Drive strength not supported

Table 17. Pad keeping enable

PKE	Description
0	Pad keeping disabled
1	Pad keeping enabled

NOTE

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

4.3 Functional description

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

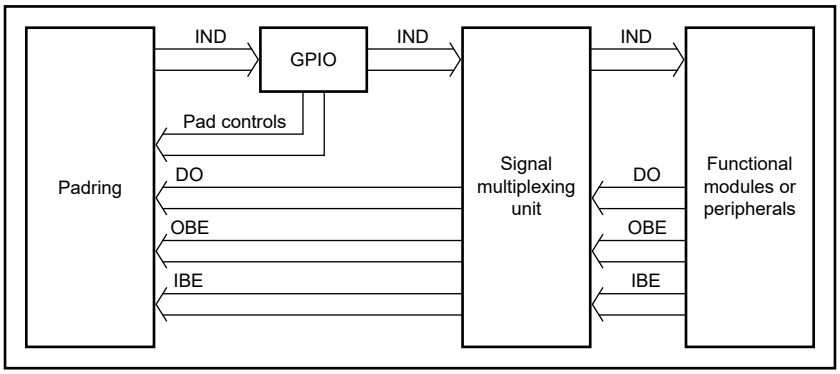


Figure 10. Signal Multiplexing

4.4 Signal Multiplexing sheet

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

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

4.4.1 IO Signal Table

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

Pin	CR	SSS	Function	Module	Description	Direction	Pad Type	I/O Power Dom	Pad State During Destructive Rese	Pad State After Destructive Reset	Pad State During Functional Reset	Pad State After Functional Reset	Pad State After Self	Pad State After FCCU Error	I/O Power Segme	MSCR [0][1][2][3]	[4][5][6]	[7][7][7]	[8][8][8]	[9][10][11]	[12][13][14]	[15][15][15]	[16][17]	[18][18]	[19][19]	[20][21]	[22][23]	[24][25]	[26][27][28]	[29][30][31]
PTA0	SIUL_MSCR0	0000_0000	GPIO[0]	SIUL		I/O	GPIO-STANDARD	VDD_HV_A	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
PTA0		0000_0001	LPSPI4_PCS2	LPSPI4	Peripheral Chip Select 2	O																							
PTA0		0000_0010	eMIOS_0_CH[17]_Y	eMIOS_0	eMIOS Channel	O																							
PTA0		0000_0011	LCU0_OUT4	LCU0	LCU Output	O																							
PTA0		0000_0100	FXIO_D2	FXIO	FlexIO Bi-directional Shift/timer I/O	O																							
PTA0		0000_0101	eMIOS_1_CH[0]_X	eMIOS_1	eMIOS Channel	O																							
PTA0		0000_0110	LPSPI0_PCS7	LPSPI0	Peripheral Chip Select 7	O																							
PTA0		0000_0111	TRGMUX_OUT3	TRGMUX	Trigger Mux Output	O																							
PTA0	-	-	ADC0_S8	ADC0	ADC Standard Input	I																							
PTA0	-	-	CMP1_IN0	CMP1	Comparator Input Signal	I																							
PTA0	SIUL_IMCR32	0000_0000	EIRQ[0]	SIUL	External Interrupt	I																							
PTA0	SIUL_IMCR32	0000_0010	eMIOS_0_CH[17]_Y	eMIOS_0	eMIOS Channel	I																							
PTA0	SIUL_IMCR59	0000_0010	eMIOS_1_CH[0]_X	eMIOS_1	eMIOS Channel	I																							
PTA0	SIUL_IMCR60	0000_0010	FXIO_D2	FXIO	FlexIO Bi-directional Shift/timer I/O	I																							
PTA0	SIUL_IMCR74	0000_0010	LPSPI0_PCS7	LPSPI0	Peripheral Chip Select 7	I																							
PTA0	SIUL_IMCR76	0000_0010	LPSPI4_PCS2	LPSPI4	Peripheral Chip Select 2	I																							
PTA0	SIUL_IMCR85	0000_0010	LPUART0_CTS	LPUART0	Clear To Send (bar)	I																							

Figure 11. IO signal table snippet

The columns of the above figure are described below:

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

Table 18. Port/MSCR mapping

Port	SIUL2_MSCRn index
PortA[0-31]	MSCR0 - MSCR31
PortB[0-31]	MSCR32 - MSCR63
PortC[0-31]	MSCR64 - MSCR95
PortD[0-31]	MSCR96 - MSCR127
PortE[0-31]	MSCR128 - MSCR159
PortF[0-31]	MSCR160 - MSCR191
PortG[0-27]	MSCR192 - MSCR219

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

- SSS: This field specifies the ALT mode of operation as per MSCR[Mux_mode]. Not all pins support all pin muxing slots. Unimplemented pin muxing slots are reserved. The corresponding pin is configured in the following pin muxing slot as follows:
  - 000: Alternative 0 (GPIO)
  - 001: Alternative 1 (chip-specific)
  - 010: Alternative 2 (chip-specific)
  - 011: Alternative 3 (chip-specific)
  - 100: Alternative 4 (chip-specific)
  - 101: Alternative 5 (chip-specific)
  - 110: Alternative 6 (chip-specific)
  - 111: Alternative 7 (chip-specific)

NOTE

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

- Function: This field specifies the functionality of the pad as per the corresponding ALT mode specified by SSS field.
- Module: The Module field contains the module which is governing the pad for the ALT mode.

- Description: This field mentions a short description of pad functionality.
- Direction: This field specifies the direction (Input, Output or Input/Output) of the pad for the concerned functionality.
- Pad Type: This field mentions the pad type of the corresponding pad.
  - GPIO-Standard:
    - Switching up to 10MHz
    - High drive-strength not supported.
    - Slew-rate control not supported.
  - GPIO-Standard plus:
    - Switching up to 25MHz
    - Supports high drive-strength.
    - Slew-rate control not supported.
  - GPIO-Medium:
    - Switching up to 50MHz
    - Supports high drive-strength.
    - Supports slew-rate control.
  - GPIO-Fast:
    - Switching up to 120MHz
    - Supports high drive-strength.
    - Supports slew-rate control.
- The next columns specify the pin number in the supported packages for the device.
- I/O Power Domain: This field refers to the power domain of associated pad (VDD_HV_A / VDD_HV_B).
- Pad State during/after Reset: These fields represent the pad states during and post different device resets.
- MSCR: This field specifies the default MSCR value for corresponding pad. Refer SIUL2_MSCRn for description of MSCR fields.
- The next two columns specify the reset value and the configurable bit fields of MSCR corresponding to pad.

4.4.2 Input muxing table

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

Destination Instance	Destination Function	CR Instance	Input CR#	Input SSS	Source Instance	Source Signal
CAN0	CAN0_RX	SIUL	SIUL_IMCR512	0000_0000	-	disable low
				0000_0001	IO_PAD	PTC2
				0000_0010	IO_PAD	PTA6
				0000_0011	IO_PAD	PTB0
				0000_0100	IO_PAD	PTA28
				0000_0101	IO_PAD	PTF21

Figure 12. Input muxing table snippet

The columns of the figure are briefly described below:

- Destination Instance: This field contains the instance name of the input path to where the signal will propagate from padring.
- Destination Function: This field mentions the function name of the input path.
- Input SSS: This field specifies the IMCR[Mux_mode] value corresponding to the pad specified in source signal column.

- Source Instance: This field specifies the source pad type. A blank is mentioned for the default source when no pad is driving the input path.
- Source Signal: This field mentions the pad name. A ‘disable low’/’disable high’ specifies the signal behavior when none of the pads are driving the input path.
- The next columns specify the pin number in the supported packages for the device.

4.4.3 MSCR/IMCR description/explanation

MSCR assignments

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

Example:

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

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

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

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

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

NOTE

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

4.4.4 Pinout diagrams

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

4.5 Pin States

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

4.5.1 Pin numbers

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

4.5.2 Power up

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

Table 19. Power up

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

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

4.5.3 Destructive Reset

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

Table 20. Destructive Reset

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

4.5.4 Functional Reset including Functional entry sequence

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

Table 21. Functional Reset including Functional entry sequence

Pins	Functional Reset Entry	Reset until SW comes up
RESET_B	LOW	LOW

Table continues on the next page...

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

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

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

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

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

Table continues on the next page...

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

Pins	Selftest	Reset until SW comes up
	ED] is Enabled and DCMRWD2[EOUT_STAT_DUR_STEST] is Enabled Else HIGH Z	ED] is Enabled and DCMRWD2[EOUT_STAT_DUR_STEST] is Enabled Else HIGH Z
CLKOUT_STANDBY	Clkout expose if DCMRWP1[CLKOUT_STANDBY] is configured to be Enabled Else HIGH Z	Clkout expose if DCMRWP1[CLKOUT_STANDBY] is configured to be Enabled Else HIGH Z
GPIOs	HIGH Z	HIGH Z

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

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

Pins	Selftest	Reset after Selftest until SW comes up
RESET_B	Same state as before selftest	Same state as before selftest
TMS,TCK,TDI,TDO as JTAG	Same state as before selftest	PULLED Values (TMS = HIGH, TCK = LOW, TDI = HIGH) TDO: No change
ETM_TRACE	Same state as before selftest	Same state as before selftest
FCCU_ERR	FCCU Error State if UTEST_MISC[FCCU_EOUT_DEDICATED] is Enabled and DCMRWD2[EOUT_STAT_DUR_STEST] is Enabled Else No change	FCCU Error State if UTEST_MISC[FCCU_EOUT_DEDICATED] is Enabled and DCMRWD2[EOUT_STAT_DUR_STEST] is Enabled Else No change
CLKOUT_STANDBY	Same state as before selftest	Same state as before selftest
GPIOs	Same state as before selftest	Same state as before selftest

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

NOTE

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

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

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

Table continues on the next page...

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

Pins	ON FCCU Fault
GPIOs	If the PAD is configured as a dedicated PAD then no change and the pad would continue performing the operation else if it is a GPIO PAD then it would behave as per the SMC configuration. • Case 1: If the SMC bit of SIUL2's MSCR register of the corresponding pin is configured as 1'b1, the pin continues its operation. • Case 2: If the SMC bit of SIUL2's MSCR register of the corresponding pin is configured as 1'b0, the pins get tristated (high-Z).

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

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

Table 25. STANDBY

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

4.6 Glossary

DSE	Drive strength enable
EXTAL	External crystal input
GPIO	General purpose input/output
IBE	Input buffer enable
INV	Invert enable
JTAG_TMS	JTAG test mode select
JTAG_TCK	JTAG test clock input
JTAG_TDI	JTAG test data input
JTAG_TDO	JTAG test data output
OBE	Output buffer enable
PUS	Pullup and pulldown select
PKE	Pad keeping enable

SMC	Safe mode control
SRC	Slew rate control
SSS	Solid-state storage
XTAL	External crystal output

Chapter 5
Cortex-M7 Overview

5.1 Introduction

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

Table 26. Cortex-M7 instances

Instances	MWCT2D17S/MWCT2D16S	MWCT2016S/MWCT2015S
CM7_0	Yes	Yes
CM7_1	Yes	No

5.1.1 Features

Cortex-M7 provides:

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

5.1.2 Related information

For detailed information on:

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

5.1.3 Buses, interconnects, and interfaces

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

Table 27. Buses and associated information

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

NOTE

MWCT2xxxS AHBP bus is enabled after reset. Therefore, accesses of all cores to on-chip peripherals are performed exclusively through this bus.

5.1.4 Core configuration

This table describes Cortex-M7 parameter settings.

Table 28. Core configuration

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

Table continues on the next page...

Table 28. Core configuration (continued)

Parameter	Configuration[1]
WIC support	Not implemented
Dual-redundant core (lock-step) CPU functionality	Not implemented
RAR	All asynchronously reset
I-cache size	Implemented [2]
D-cache size	Implemented [2]
ITCM	• All chips: 32 KB for CM7_0/1, 64 KB for CM7_2
DTCM0	• All chips: 32 KB for CM7_0/1, 64 KB for CM7_2
DTCM1	• All chips: 32 KB for CM7_0/1, 64 KB for CM7_2

1. Armv7-M (Harvard architecture), six-stage superscalar plus branch prediction
2. See chapter 'Memory and Memory Interfaces' for details

5.2 Speculative accesses

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

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

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

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

- Speculative access to an uninitialized RAM memory location that causes a double bit error {the related fault processing by the FCCU is performed independently from the processor}.
- Speculative access to a peripheral that causes an unwanted operation; for example, a FIFO read {the corresponding data may be removed from the FIFO without being processed}.
- Speculative access to a peripheral that is not clocked, powered down, or cannot respond {the access may not be terminated, resulting an access that is stalled, system blockage}.
- Speculative access to an address range that causes an unexpected error being reported; for example, a read-while-write error of the embedded flash (indicated by setting MCRS[RWE]) during a flash erase or program operation {which may hide real errors}.

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

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

- Speculative reads are never made on the AHBP interface.
- Speculative writes are never made.

Memory regions mapped to a TCM are always treated as Normal Memory (equivalent to the MPU attribute) and are therefore always subject to speculation. Related issues can be avoided by properly initializing any TCM memory before a corresponding access may occur.

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

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

5.3 Debug facilities

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

5.4 Vector fetch behavior on Cortex-M7

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

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

5.5 TCM retry

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

5.6 Glossary

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

Chapter 6
Miscellaneous Control Module (MCM)

6.1 Chip-specific MCM information

6.1.1 MCM instances and configuration

This chip supports up to four instances of MCM:

- MCM_0
- MCM_1
- MCM_2
- MCM_3

Table 29. MCM instances

Instances	MWCT2D16S/ MWCT2D17S	MWCT2016S/MWCT2015S
MCM_0	Yes	Yes
MCM_1	Yes	Yes

Table 30. Memories for all chips

Memory	MWCT2015S/MWCT2016S (Single core)	MWCT2D16S/MWCT2D17S(Dual core)
Icache	8 KB	8 KB (per core)
Dcache	8 KB	8 KB (per core)
ITCM	32 KB	32 KB (per core)
DTCM	64 KB	64 KB (per core)

6.2 Introduction

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

6.2.1 Features

The MCM includes the following features:

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

6.3 Functional description

6.3.1 Interrupts

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

- FPU input denormal interrupt is enabled (FIDCE) and an input is denormalized (FIDC).
- FPU inexact interrupt is enabled (FIXCE) and a number is inexact (FIXC).
- FPU underflow interrupt is enabled (FUFCE) and an underflow occurs (FUFC).
- FPU overflow interrupt is enabled (FOFCE) and an overflow occurs (FOFC).
- FPU divide-by-zero interrupt is enabled (FDZCE) and a divide-by-zero occurs (FDZC).
- FPU invalid operation interrupt is enabled (FIOCE) and an invalid operation occurs (FIOC).
- Write abort interrupt is enabled (WABE) and a write abort occurs (CORTEX-M7 WABORTS INDICATOR).

Determining interrupt source

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

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

6.4 Memory map and register descriptions

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

NOTE

The following actions result in bus errors:

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

6.4.1 MCM register descriptions

6.4.1.1 MCM memory map

MCM_0_CM7 base address: E008_0000h

MCM_1_CM7 base address: E008_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	SoC-defined Platform Revision (PLREV)	16	R	0000h
2h	Processor Core Type (PCT)	16	R	AC70h
Ch	Core Platform Control (CPCR)	32	RW	0000_0200h
10h	Interrupt Status and Control (ISCR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
30h	Processor Identifier (PID)	8	RW	00h
400h	Local Memory Descriptor 0 (LMEM_DESC_0)	32	RW	8606_0000h
404h - 408h	Local Memory Descriptor a (LMEM_DESC_1 - LMEM_DESC_2)	32	RW	8604_2000h
40Ch	Local Memory Descriptor 3 (LMEM_DESC_3)	32	RW	8426_4000h
410h	Local Memory Descriptor 4 (LMEM_DESC_4)	32	RW	8444_6000h

6.4.1.2 SoC-defined Platform Revision (PLREV)

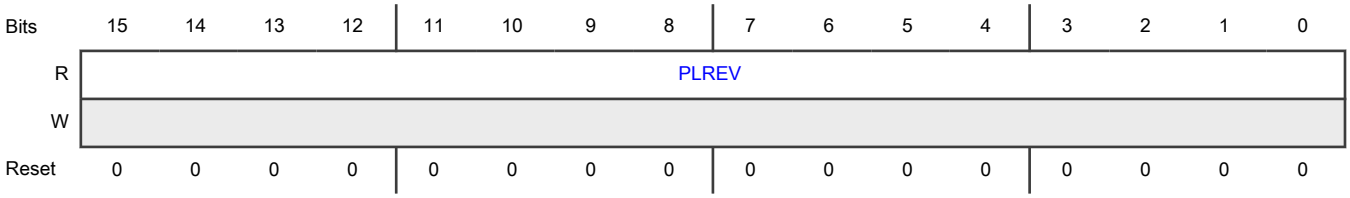
Offset

Register	Offset
PLREV	0h

Function

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

Diagram



Fields

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

6.4.1.3 Processor Core Type (PCT)

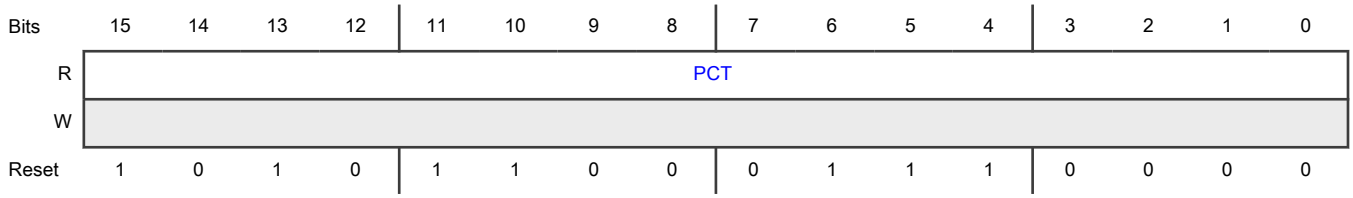
Offset

Register	Offset
PCT	2h

Function

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

Diagram



Fields

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

6.4.1.4 Core Platform Control (CPCR)

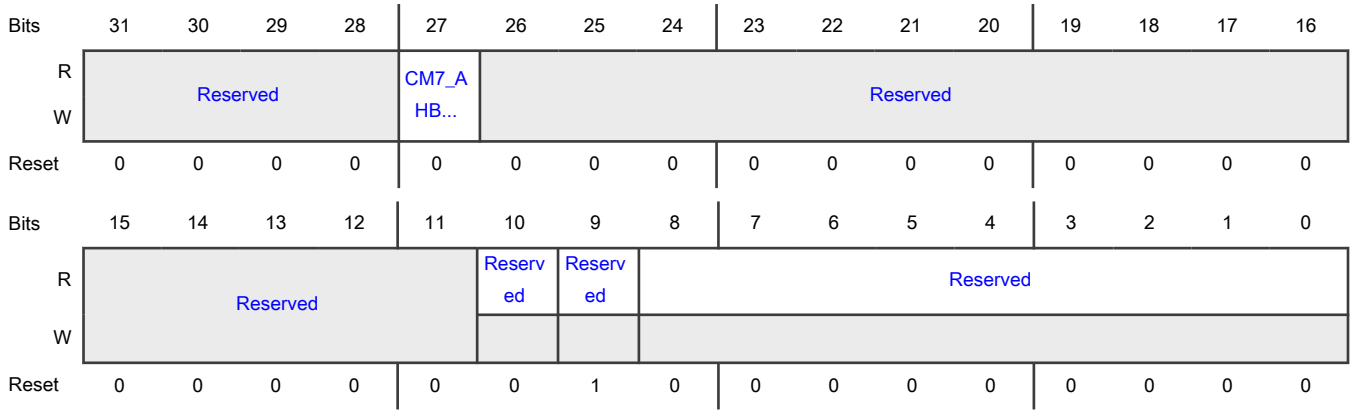
Offset

Register	Offset
CPCR	Ch

Function

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

Diagram



Fields

Field	Function
31-28 —	Reserved
27 CM7_AHBSPRI	AHB Slave Priority Indicates the access priority on the AHBS port of the Cortex-M7. **NOTE** This setting has no effect unless enabled by AHBSCR[CTL][1] of the Cortex-M7 core. 0b - Uses a round-robin arbitration scheme 1b - AHB-slave access has priority over a core access
26-11 —	Reserved
10 —	Reserved
9 —	Reserved
8-0 —	Reserved

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

6.4.1.5 Interrupt Status and Control (ISCR)

Offset

Register	Offset
ISCR	10h

Function

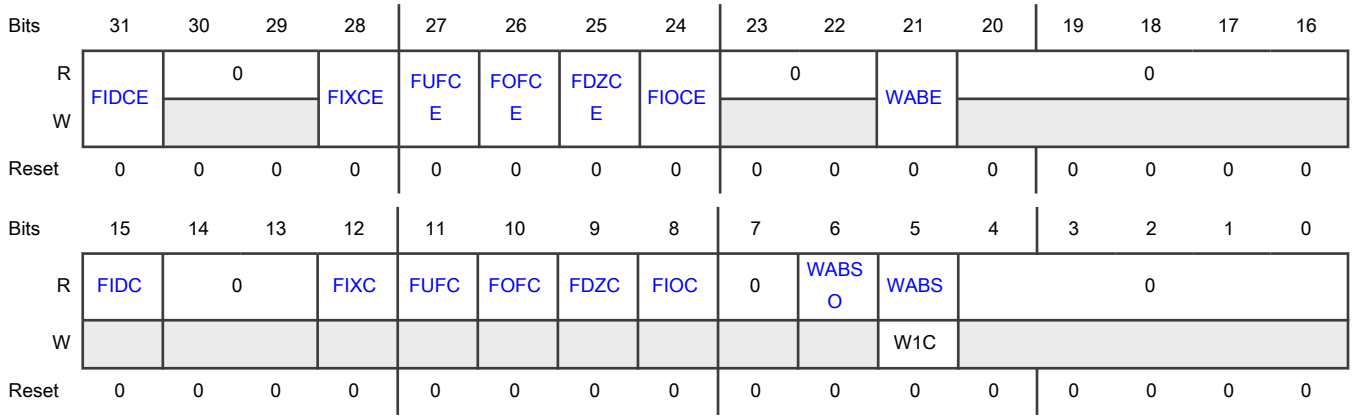
Defines the configuration and reports the status for a number of core-related interrupt exception conditions. It includes:

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

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

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

Diagram



Fields

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

Table continues on the next page...

Table continued from the previous page...

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

Table continues on the next page...

Table continued from the previous page...

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

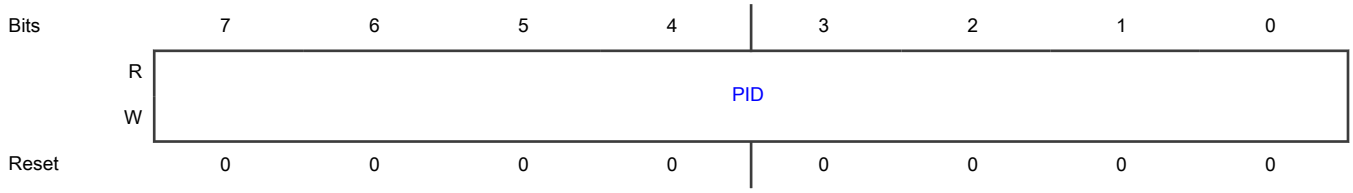
6.4.1.6 Processor Identifier (PID)

Offset

Register	Offset
PID	30h

Function
Contains the CPU process ID.

Diagram



Fields

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

6.4.1.7 Local Memory Descriptor 0 (LMEM_DESC_0)

Offset

Register	Offset
LMEM_DESC_0	400h

Function

NOTE

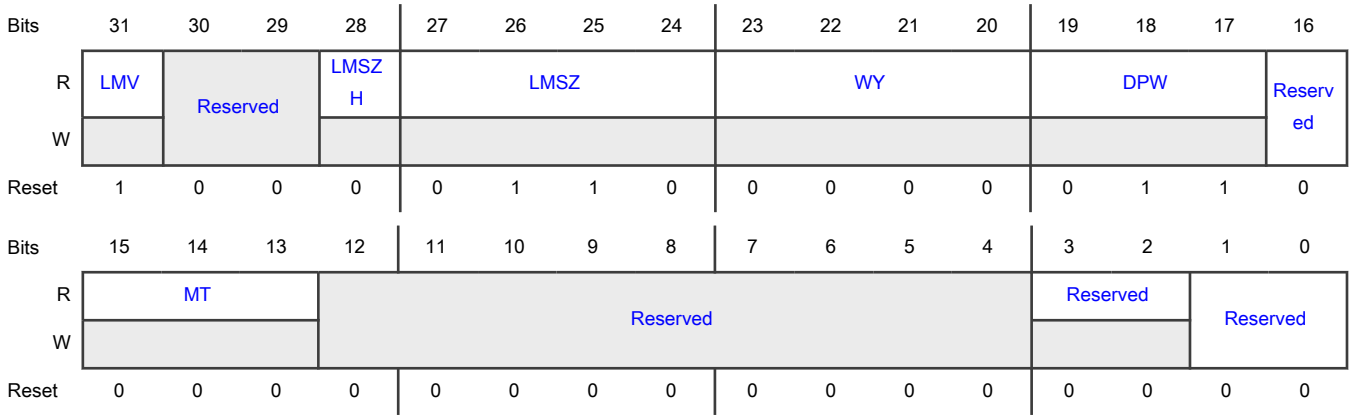
The DESC_a registers map to the LMEMs in this way:

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

NOTE

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

Diagram



Fields

Field	Function
31 LMV	Local Memory Valid This read-only field defines the validity (presence) of the local memory. 0b - LMEM*n* not present 1b - LMEM*n* present
30-29 —	Reserved
28 LMSZH	LMEM Size Hole Used for local memories that are not fully populated (that is, local memories that include a memory "hole" in the upper 25 % of the address range). 0b - LMEM*n* is a power-of-2 capacity 1b - LMEM*n* is not a power-of-2, with capacity of 0.75 × LMSZ
27-24 LMSZ	Local Memory Size 0000b - 0 KB 0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB 0111b - 64 KB 1000b - 128 KB 1001b - 256 KB

Table continues on the next page...

Table continued from the previous page...

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

6.4.1.8 Local Memory Descriptor a (LMEM_DESC_1 - LMEM_DESC_2)

Offset

Register	Offset
LMEM_DESC_1	404h
LMEM_DESC_2	408h

Function

NOTE

The DESC_a registers map to the LMEMs in this way:

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

NOTE

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

Diagram

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

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

Fields

Field	Function
31 LMV	Local Memory Valid This read-only field defines the validity (presence) of the local memory. 0b - LMEM*n* not present 1b - LMEM*n* present

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-29 —	Reserved
28 LMSZH	LMEM Size Hole Used for local memories that are not fully populated (that is, local memories that include a memory "hole" in the upper 25 % of the address range). 0b - LMEM*n* is a power-of-2 capacity 1b - LMEM*n* is not a power-of-2, with capacity of 0.75 × LMSZ
27-24 LMSZ	Local Memory Size 0000b - 0 KB 0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB 0111b - 64 KB 1000b - 128 KB 1001b - 256 KB 1010b - 512 KB 1011b - 1024 KB 1100b - 2048 KB 1101b - 4096 KB 1110b - 8192 KB 1111b - 16384 KB
23-20 WY	Level 1 Cache Ways 0000b - No cache 0010b - 2-way set associative 0100b - 4-way set associative
19-17 DPW	Data Path Width Defines the LMEM*n* data path width, which is the width of the local memory. 000b-001b - Reserved 010b - 32 bits

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - 64 bits 100b-111b - Reserved
16 —	Reserved
15-13 MT	Memory Type 000b - ITCM 001b - DTCM 010b - I-cache 011b - D-cache
12-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

6.4.1.9 Local Memory Descriptor 3 (LMEM_DESC_3)

Offset

Register	Offset
LMEM_DESC_3	40Ch

Function

NOTE

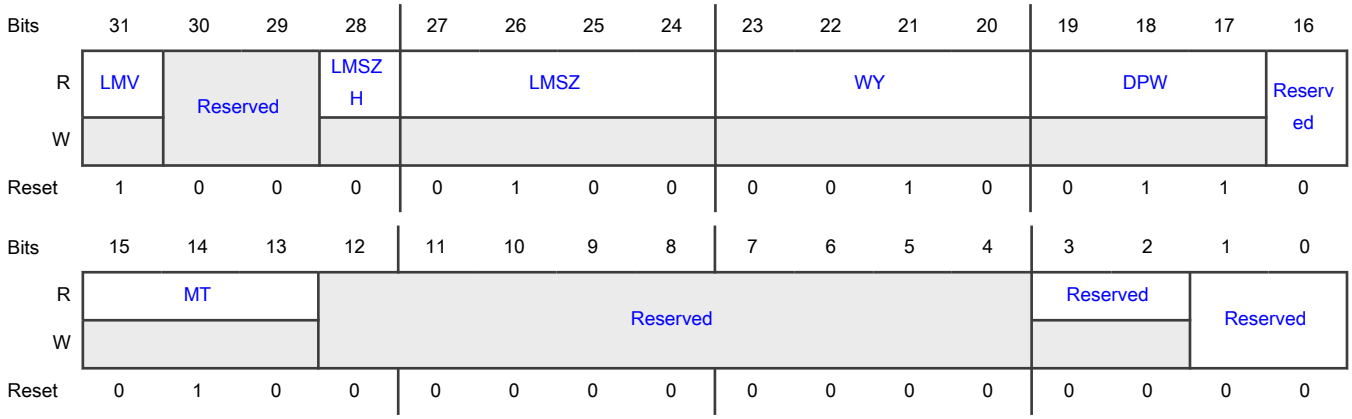
The DESC_a registers map to the LMEMs in this way:

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

NOTE

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

Diagram



Fields

Field	Function
31 LMV	Local Memory Valid This read-only field defines the validity (presence) of the local memory. 0b - LMEM*n* not present 1b - LMEM*n* present
30-29 —	Reserved
28 LMSZH	LMEM Size Hole Used for local memories that are not fully populated (that is, local memories that include a memory "hole" in the upper 25 % of the address range). 0b - LMEM*n* is a power-of-2 capacity 1b - LMEM*n* is not a power-of-2, with capacity of 0.75 × LMSZ
27-24 LMSZ	Local Memory Size 0000b - 0 KB 0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB 0111b - 64 KB 1000b - 128 KB 1001b - 256 KB

Table continues on the next page...

Table continued from the previous page...

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

6.4.1.10 Local Memory Descriptor 4 (LMEM_DESC_4)

Offset

Register	Offset
LMEM_DESC_4	410h

Function

NOTE

The DESC_a registers map to the LMEMs in this way:

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

NOTE

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

Diagram

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

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

Fields

Field	Function
31 LMV	Local Memory Valid This read-only field defines the validity (presence) of the local memory. 0b - LMEM*n* not present 1b - LMEM*n* present
30-29	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
28 LMSZH	LMEM Size Hole Used for local memories that are not fully populated (that is, local memories that include a memory "hole" in the upper 25 % of the address range). 0b - LMEM*n* is a power-of-2 capacity 1b - LMEM*n* is not a power-of-2, with capacity of 0.75 × LMSZ
27-24 LMSZ	Local Memory Size 0000b - 0 KB 0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB 0111b - 64 KB 1000b - 128 KB 1001b - 256 KB 1010b - 512 KB 1011b - 1024 KB 1100b - 2048 KB 1101b - 4096 KB 1110b - 8192 KB 1111b - 16384 KB
23-20 WY	Level 1 Cache Ways 0000b - No cache 0010b - 2-way set associative 0100b - 4-way set associative
19-17 DPW	Data Path Width Defines the LMEM*n* data path width, which is the width of the local memory. 000b-001b - Reserved 010b - 32 bits 011b - 64 bits 100b-111b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 —	Reserved
15-13 MT	Memory Type 000b - ITCM 001b - DTCM 010b - I-cache 011b - D-cache
12-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

6.5 Glossary

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

Chapter 7
Miscellaneous System Control Module (MSCM)

7.1 Chip-specific MSCM information

7.1.1 MSCM instance

This chip has one instance of MSCM.

NOTE

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

XN_CTRL register is reserved for MWCT2D17S.

7.1.2 Reporting of core-to-core interrupts

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

7.2 Overview

MSCM contains registers for:

- CPU configuration
- On-chip memory control
- Interrupt router control
- Message-based interrupt configuration
- Virtual management

7.2.1 Features

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

7.3 Functional description

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

7.3.1 MSI routing

MSIs are interrupts that are indirectly broadcast to a target core by writing configuration bits in MSCM. These MSIs can be initiated by one core targeting another core in the system (known as core-to-core interrupts). These MSIs are initiated via writes to the IRCP0IGR0 register and managed through the IRCP0ISR0 register, where *n* indicates the logical core number (0-1) and *m* represents the interrupt number (0-3). The Cortex-M7 cores can support up to four outstanding core-to-core interrupts.

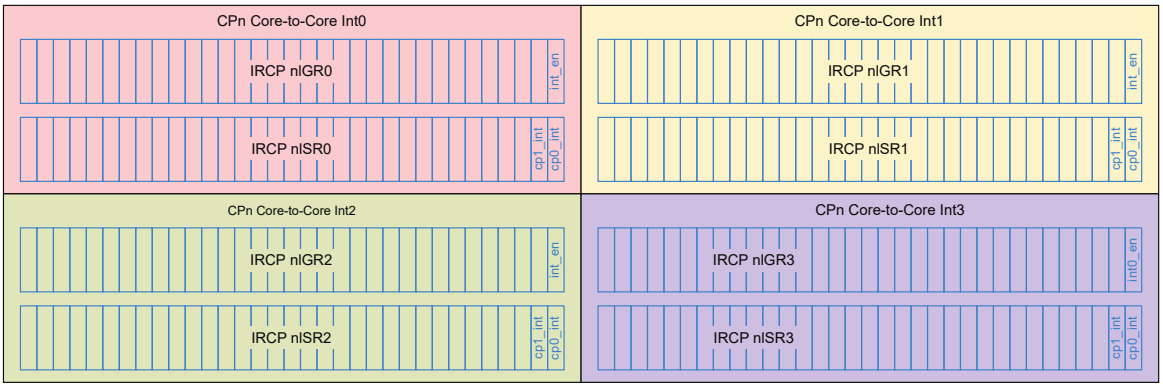


Figure 13. IRCP*n*IGR*m*/IRCP*n*ISR*m* pairs for one core

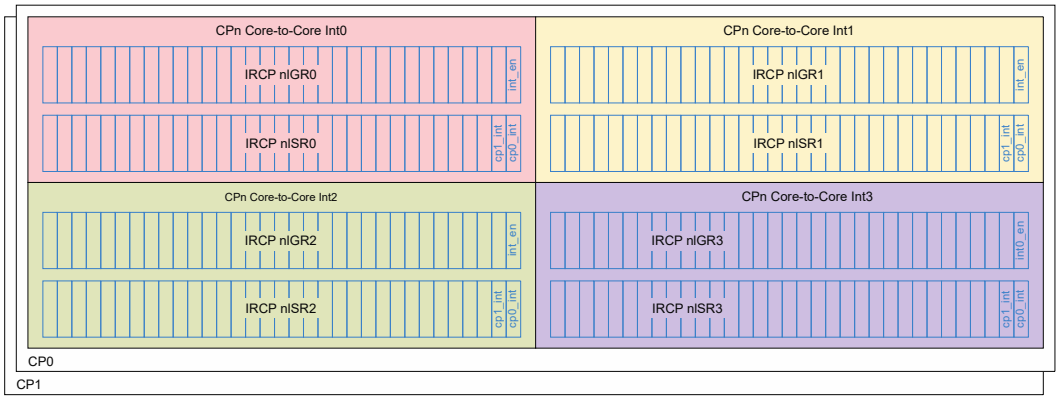


Figure 14. IRCP*n*IGR*m*/IRCP*n*ISR*m* pairs per core

7.3.1.1 Core-to-core MSIs

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

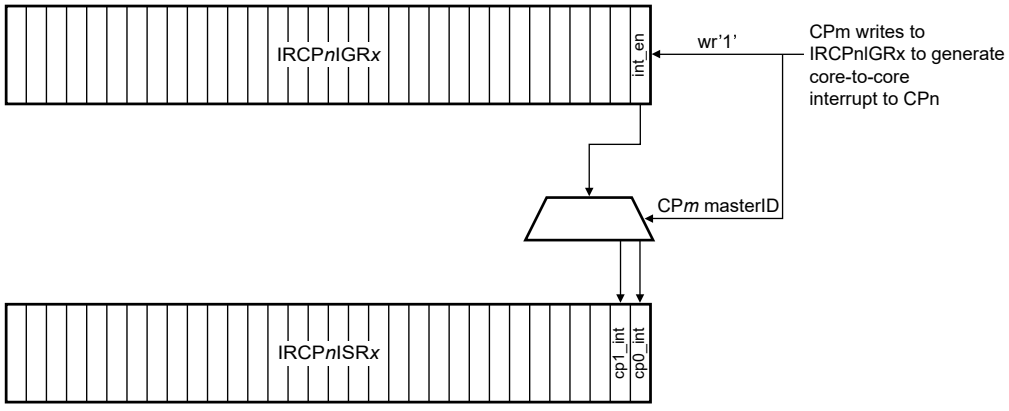


Figure 15. Initiating a core-to-core MSI via IRCP*n*IGR*x*/IRCP*n*ISR*x*

7.3.2 Interrupt steering and semaphores

7.3.2.1 Interrupt handling overview

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

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

7.3.2.2 MSCM interrupt router functional description

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

7.3.3 Clocking

This module has no clocking considerations.

7.4 External signals

This module has no external signals.

7.5 Initialization

This module does not require initialization.

7.6 Memory map and register definition

7.6.1 Core configuration registers

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

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

Table 31. MSCM core configuration partitioning

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

NOTE

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

7.6.2 Shared peripheral interrupt (SPI) routing

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

7.6.3 MSCM register descriptions

7.6.3.1 MSCM memory map

MSCM base address: 4026_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Processor X Type (CPXTYPE)	32	R	See section
4h	Processor X Number (CPXNUM)	32	R	See section
8h	Processor X Revision (CPXREV)	32	R	See section
Ch	Processor X Configuration 0 (CPXCFG0)	32	R	0502_0504h
10h	Processor X Configuration 1 (CPXCFG1)	32	R	See section
14h	Processor X Configuration 2 (CPXCFG2)	32	R	See section
18h	Processor X Configuration 3 (CPXCFG3)	32	R	0000_000Bh
20h	Processor 0 Type (CP0TYPE)	32	R	434D_3730h
24h	Processor 0 Number (CP0NUM)	32	R	See section
28h	Processor 0 Count (CP0REV)	32	R	See section
2Ch	Processor 0 Configuration 0 (CP0CFG0)	32	R	0502_0504h
30h	Processor 0 Configuration 1 (CP0CFG1)	32	R	0000_0000h
34h	Processor 0 Configuration 2 (CP0CFG2)	32	R	See section
38h	Processor 0 Configuration 3 (CP0CFG3)	32	R	0000_000Bh
40h	Processor 1 Type (CP1TYPE)	32	R	434D_3731h
44h	Processor 1 Number (CP1NUM)	32	R	0000_0001h
48h	Processor 1 Count (CP1REV)	32	R	See section
4Ch	Processor 1 Configuration 0 (CP1CFG0)	32	R	0502_0504h
50h	Processor 1 Configuration 1 (CP1CFG1)	32	R	See section
54h	Processor 1 Configuration 2 (CP1CFG2)	32	R	See section
58h	Processor 1 Configuration 3 (CP1CFG3)	32	R	0000_000Bh
400h	Interrupt Router Configuration (IRCPCFG)	32	RW	0000_0000h
600h	Enable Interconnect Error Detection (ENEDC)	32	RW	0000_0000h
700h	AHB Gasket Configuration (IAHBCFGREG)	32	RW	0000_0000h
880h - A5Eh	Interrupt Router Shared Peripheral Routing Control (IRSPRC0 - IRSPRC239)	16	RW	0003h

7.6.3.2 Processor X Type (CPXTYPE)

Offset

Register	Offset
CPXTYPE	0h

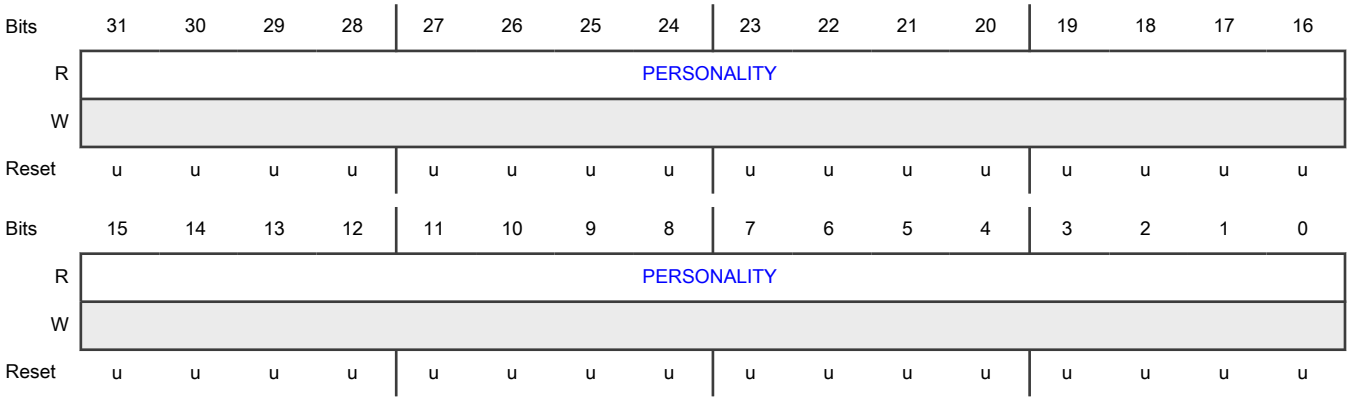
Function

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

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

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-0 PERSONALITY	Personality of CPx Defines the processor personality for CP*x*.

Processor	Personality
CP*x* = Cortex-M7_0	43_4D_37_30h
CP*x* = Cortex-M7_3	43_4D_37_33h

7.6.3.3 Processor X Number (CPXNUM)

Offset

Register	Offset
CPXNUM	4h

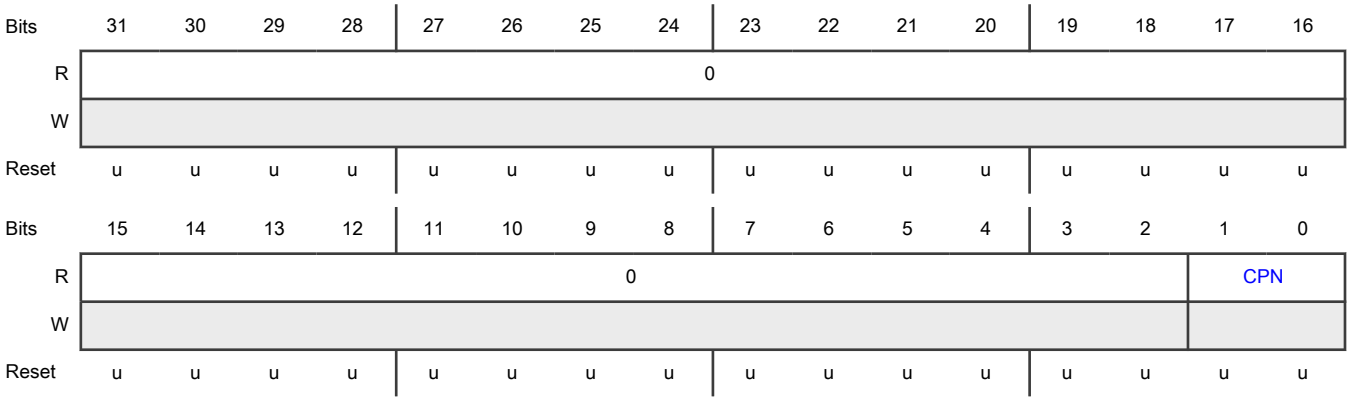
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

7.6.3.4 Processor X Revision (CPXREV)

Offset

Register	Offset
CPXREV	8h

Function

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

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

Access: User or privileged read-only

Diagram

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

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

Fields

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

7.6.3.5 Processor X Configuration 0 (CPXCFG0)

Offset

Register	Offset
CPXCFG0	Ch

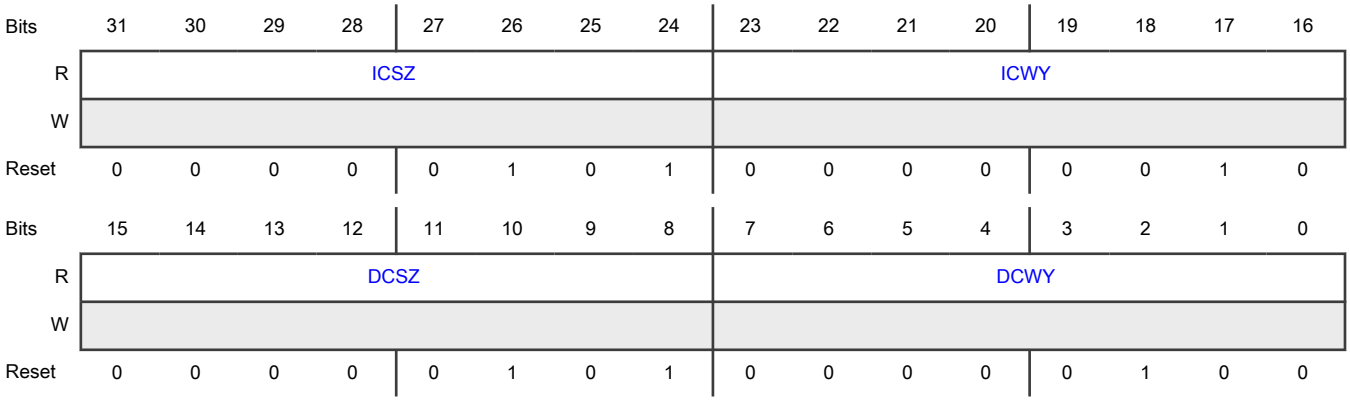
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

7.6.3.6 Processor X Configuration 1 (CPXCFG1)

Offset

Register	Offset
CPXCFG1	10h

Function

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

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

Access: User or privileged read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	L2SZ								L2WY							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

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

Fields

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

7.6.3.7 Processor X Configuration 2 (CPXCFG2)

Offset

Register	Offset
CPXCFG2	14h

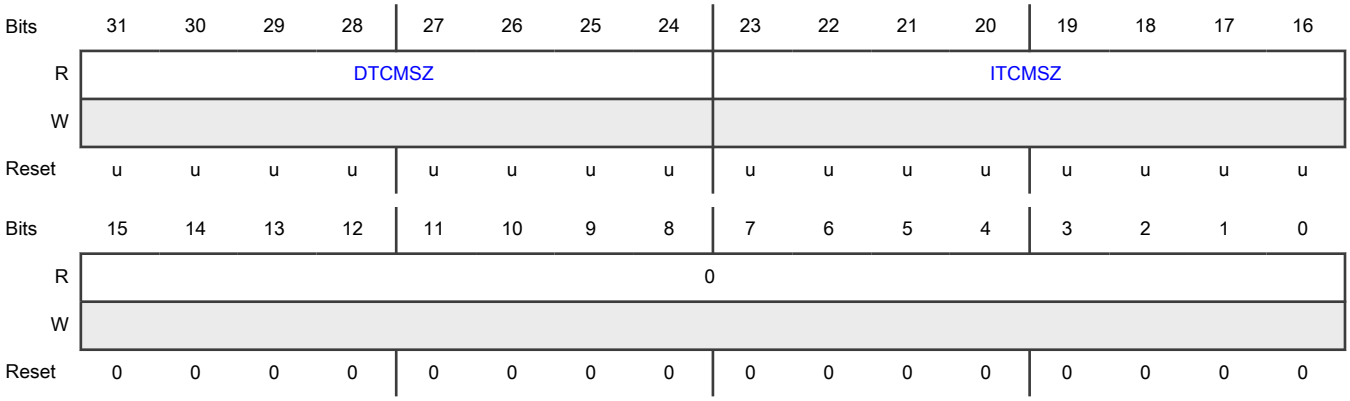
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• ITCMSZ = 07 in Decoupled mode (32 KB) • ITCMSZ = 08 for Cortex-M7_0 in Lockstep mode (64 KB)
15-0 —	Reserved

7.6.3.8 Processor X Configuration 3 (CPXCFG3)

Offset

Register	Offset
CPXCFG3	18h

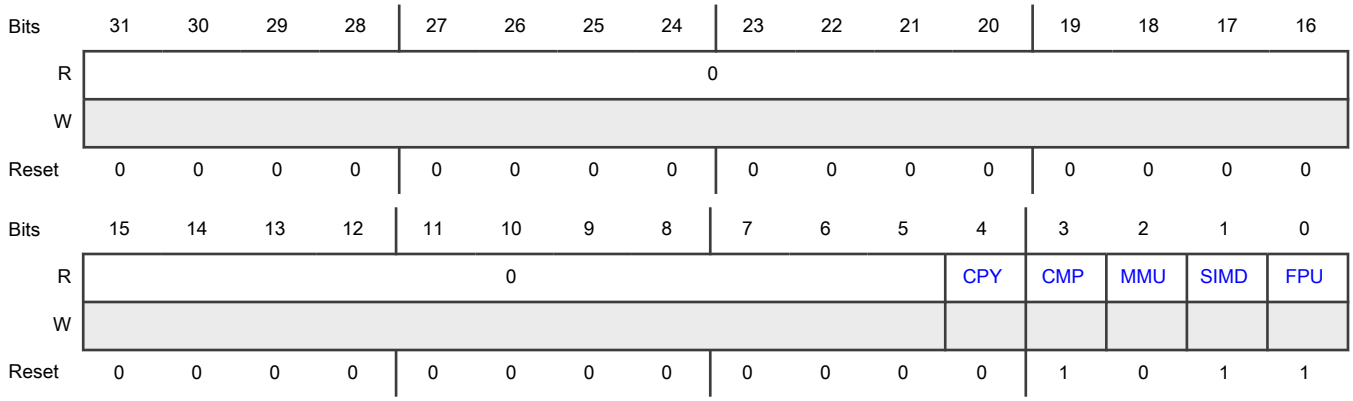
Function

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

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

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-5 —	Reserved
4	Cryptography

Table continues on the next page...

Table continued from the previous page...

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

7.6.3.9 Processor 0 Type (CP0TYPE)

Offset

Register	Offset
CP0TYPE	20h

Function

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

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

Access: User or privileged read-only

Diagram

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

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

Fields

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

7.6.3.10 Processor 0 Number (CP0NUM)

Offset

Register	Offset
CP0NUM	24h

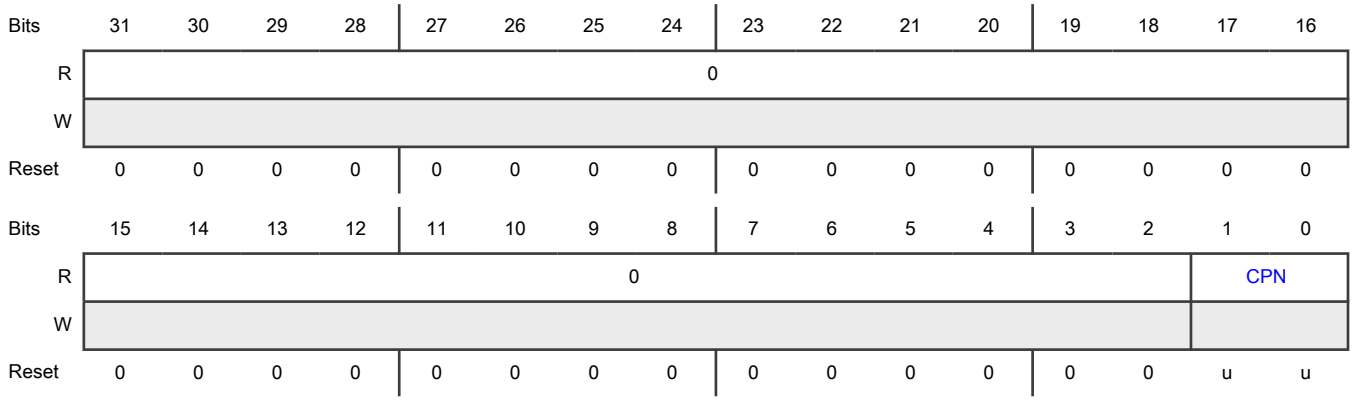
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

7.6.3.11 Processor 0 Count (CP0REV)

Offset

Register	Offset
CP0REV	28h

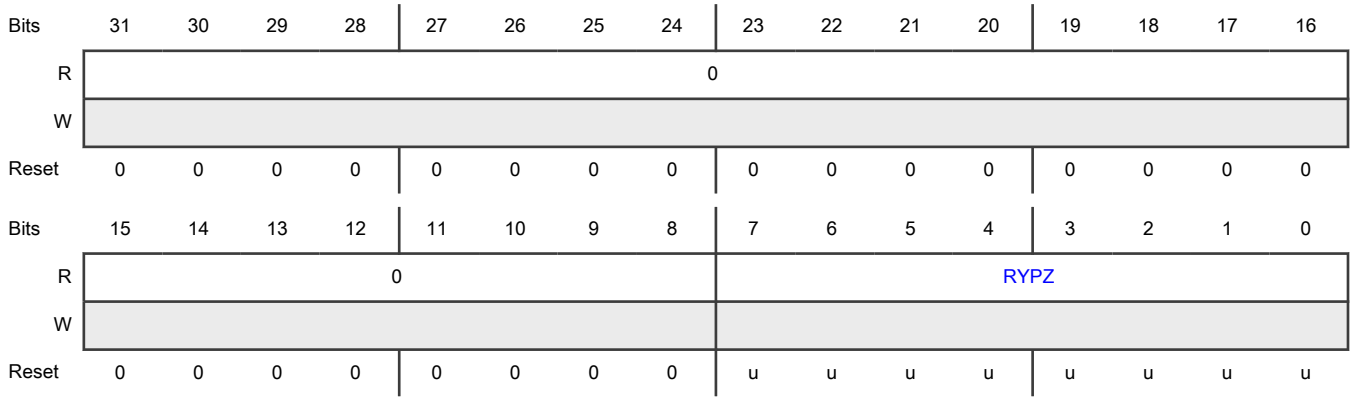
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

7.6.3.12 Processor 0 Configuration 0 (CP0CFG0)

Offset

Register	Offset
CP0CFG0	2Ch

Function

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

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

Access: User or privileged read-only

Diagram

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

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

Fields

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

7.6.3.13 Processor 0 Configuration 1 (CP0CFG1)

Offset

Register	Offset
CP0CFG1	30h

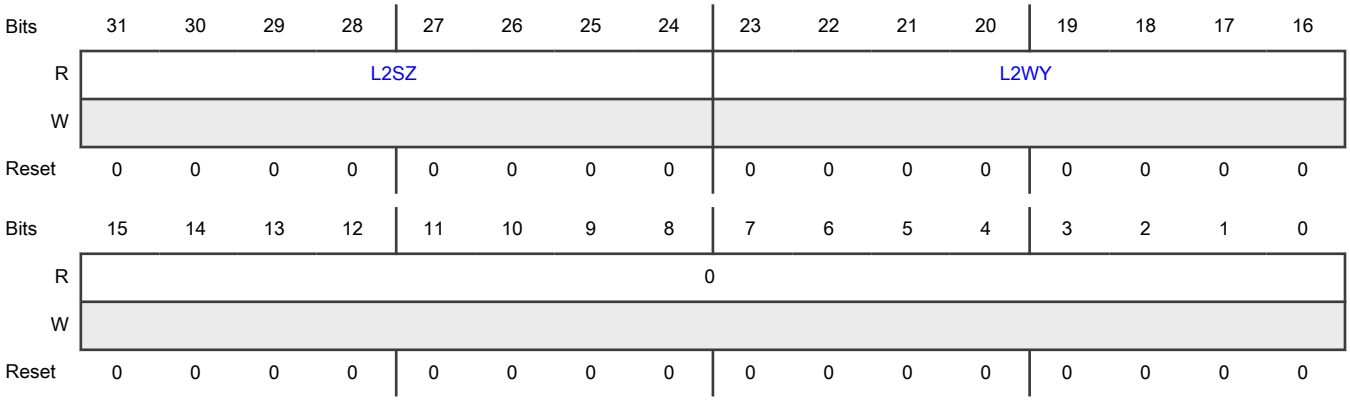
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

7.6.3.14 Processor 0 Configuration 2 (CP0CFG2)

Offset

Register	Offset
CP0CFG2	34h

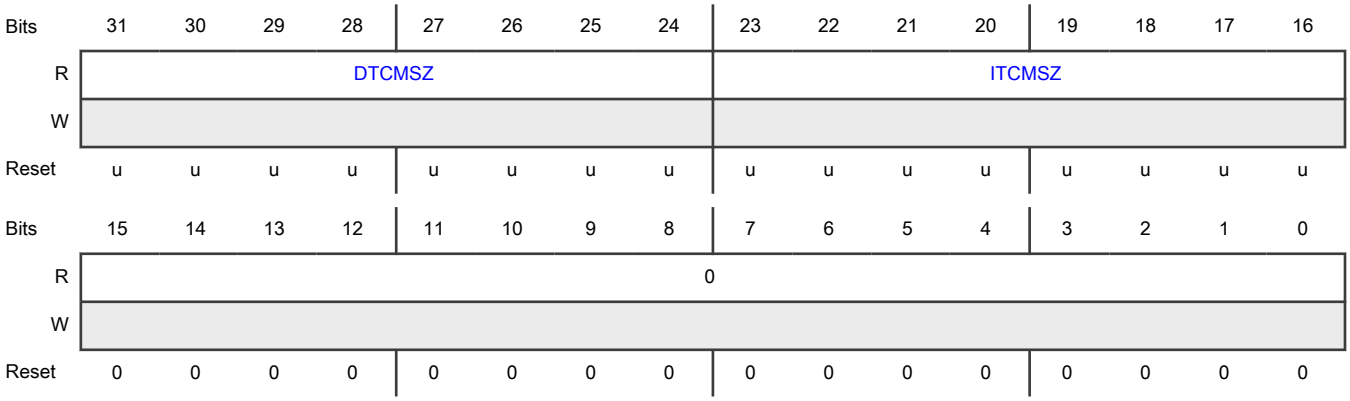
Function

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

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

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-24 DTCMSZ	Tightly Coupled Data Memory Size Provides an encoded value of the tightly coupled local data memory size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, TMLSZ is a nonzero value, and TMLSZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip: • DTCMSZ = 8h in Decoupled mode (64 KB) • DTCMSZ = 9h in Lockstep mode (128 KB)
23-16 ITCMSZ	Instruction Tightly Coupled Memory Size Provides an encoded value of the tightly coupled local instruction memory size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, TMUSZ is a nonzero value, and TMUSZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip: • ITCMSZ = 7h in Decoupled mode (32 KB) • ITCMSZ = 8h in Lockstep mode (64 KB)
15-0 —	Reserved

7.6.3.15 Processor 0 Configuration 3 (CP0CFG3)

Offset

Register	Offset
CP0CFG3	38h

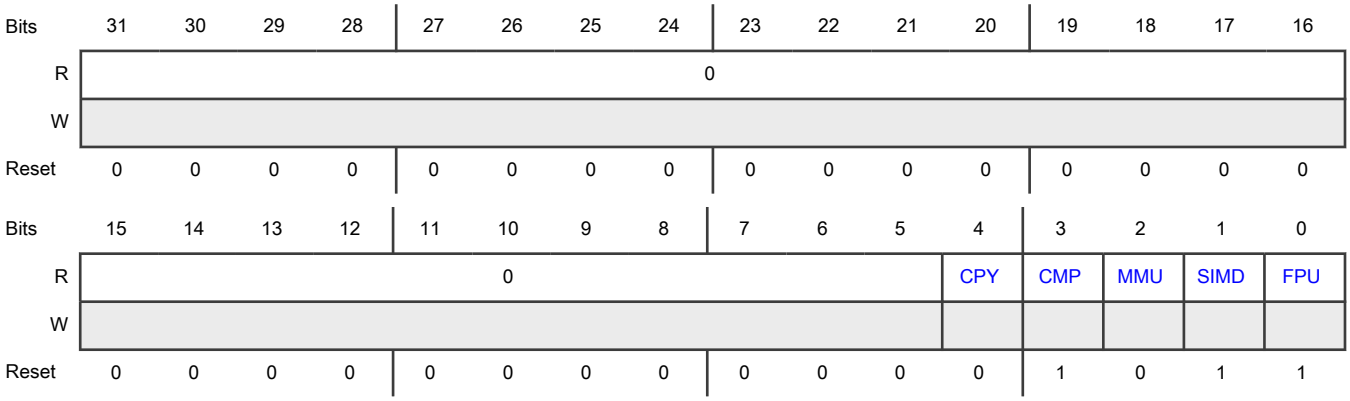
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

Table continues on the next page...

Table continued from the previous page...

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

7.6.3.16 Processor 1 Type (CP1TYPE)

Offset

Register	Offset
CP1TYPE	40h

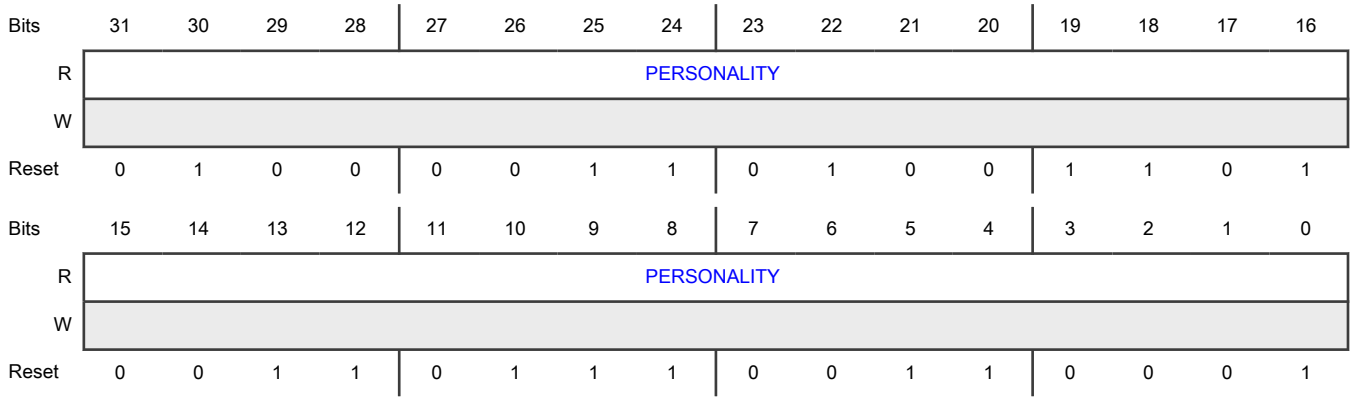
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

7.6.3.17 Processor 1 Number (CP1NUM)

Offset

Register	Offset
CP1NUM	44h

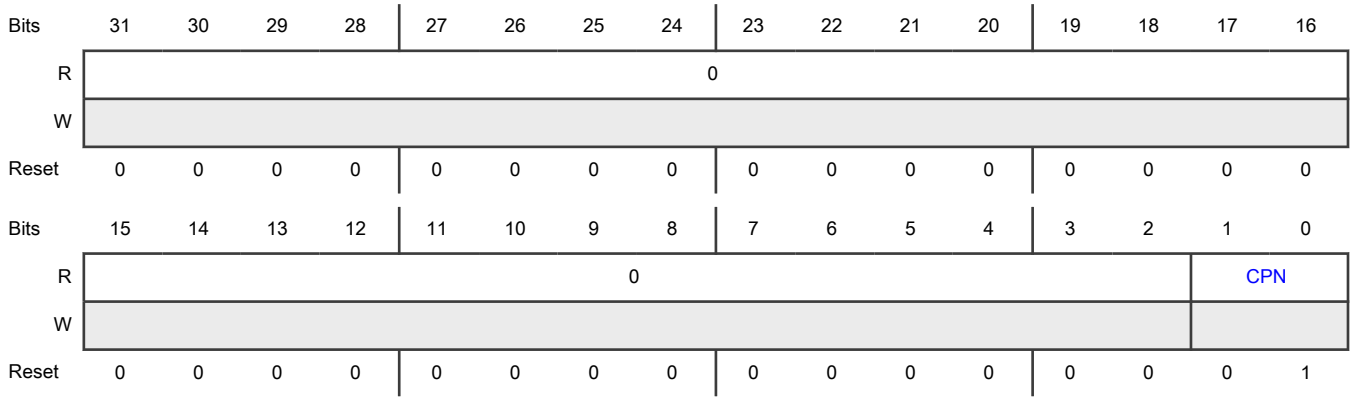
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

7.6.3.18 Processor 1 Count (CP1REV)

Offset

Register	Offset
CP1REV	48h

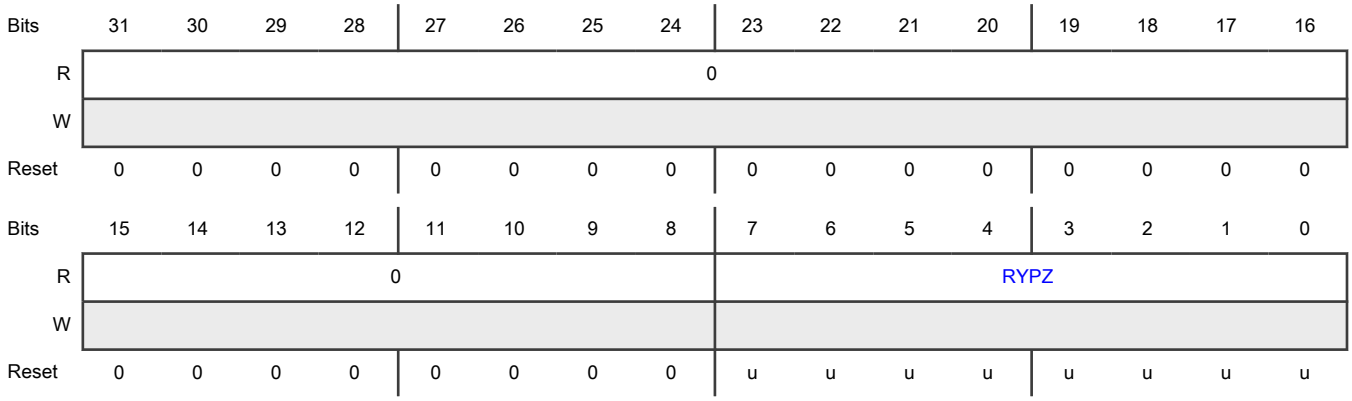
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

7.6.3.19 Processor 1 Configuration 0 (CP1CFG0)

Offset

Register	Offset
CP1CFG0	4Ch

Function

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

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

Access: User or privileged read-only

Diagram

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

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

Fields

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

7.6.3.20 Processor 1 Configuration 1 (CP1CFG1)

Offset

Register	Offset
CP1CFG1	50h

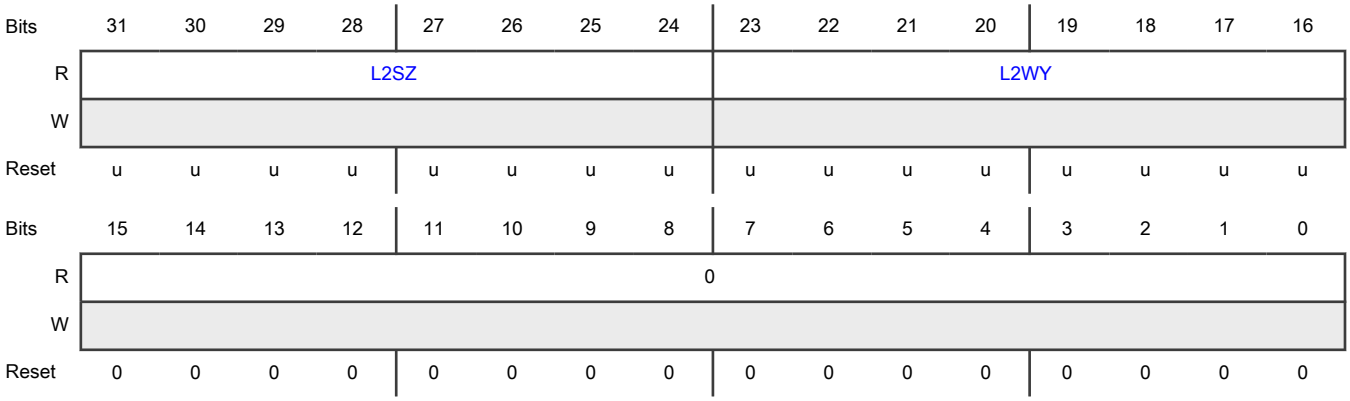
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

7.6.3.21 Processor 1 Configuration 2 (CP1CFG2)

Offset

Register	Offset
CP1CFG2	54h

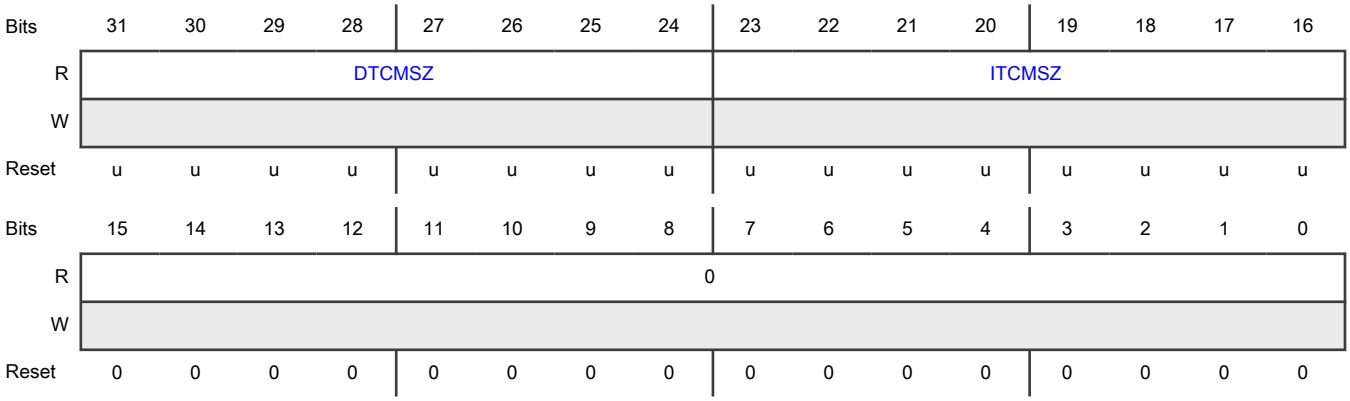
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

7.6.3.22 Processor 1 Configuration 3 (CP1CFG3)

Offset

Register	Offset
CP1CFG3	58h

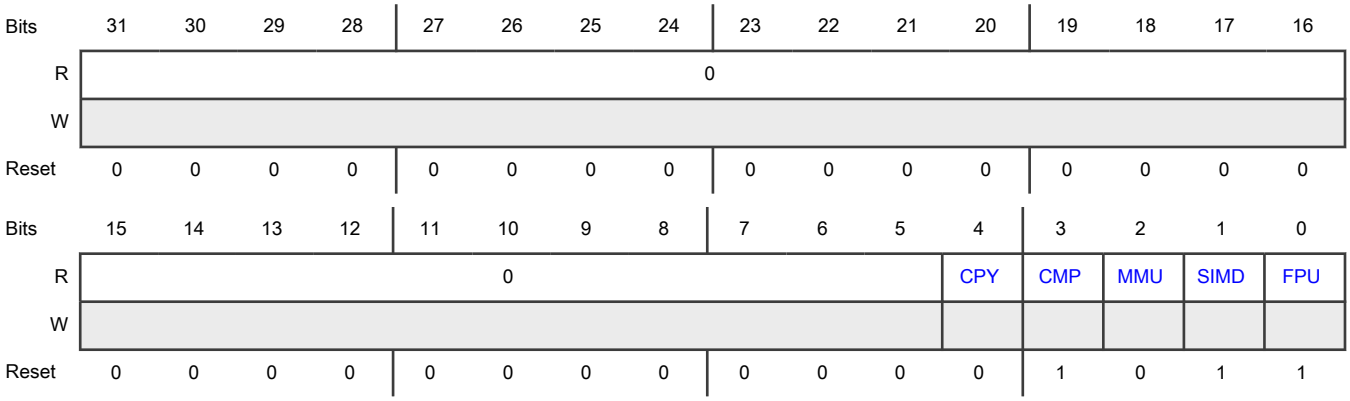
Function

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

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

Access: User or privileged read-only

Diagram



Fields

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

Table continues on the next page...

Table continued from the previous page...

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

7.6.3.23 Interrupt Router Configuration (IRCPCFG)

Offset

Register	Offset
IRCPCFG	400h

Function

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

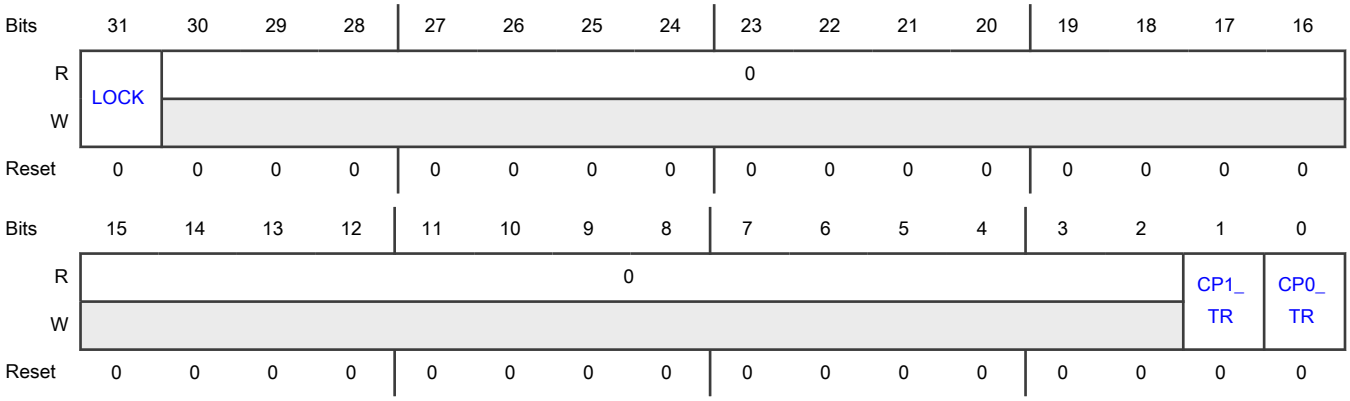
Privileged, 32-bit accesses from the:

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

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

Access: Privileged mode only

Diagram



Fields

Field	Function
31 LOCK	Lock Provides a locking mechanism that can be used to limit the ability to write to the register. After you write 1 to this field, it remains 1 until the next reset. 0b - Register can be written by any privileged write 1b - Register is locked (read-only) until the next reset
30-2 —	Reserved
1 CP1_TR	CP1 as Trusted Core Indicates whether CP1 is a trusted core with access to read the full contents of IRCP*n*ISR*m*. 0b - Not trusted 1b - Trusted
0 CP0_TR	CP0 as Trusted Core Indicates whether CP0 is a trusted core with access to read the full contents of IRCP*n*ISR*m*. 0b - Not trusted 1b - Trusted

7.6.3.24 Enable Interconnect Error Detection (ENEDC)

Offset

Register	Offset
ENEDC	600h

Function

Enables interconnect error detection.

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

Access: Privileged mode only

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EN_ADD_...	EN_WR_P...	EN_ADD_...	EN_WR_P...	EN_ADD_...	EN_ADD_...	EN_ADD_...	EN_RD_T...	EN_RD_C...	EN_RD_C...	EN_RD_E...	EN_RD_H...	0	EN_RD_E...	EN_RD_C...	EN_RD_C...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 EN_ADD_CM7_1_TCM	Enable Address Check Cortex-M7_1_TCM Enables the address check for Cortex-M7_1_TCM backdoor path. 0b - Disable 1b - Enable
28 EN_WR_CM7_1_TCM	Enable Write Data Check Cortex-M7_1_TCM Enables the write data check for Cortex-M7_1_TCM backdoor path. 0b - Disable 1b - Enable
27 EN_ADD_CM7_0_TCM	Enable Address Check Cortex-M7_0_TCM Enables the address check for Cortex-M7_0_TCM backdoor path. 0b - Disable 1b - Enable
26 EN_WR_CM7_0_TCM	Enable Write Data Check Cortex-M7_0_TCM Enables the write data check for Cortex-M7_0_TCM backdoor path. 0b - Disable 1b - Enable
25	Enable Address Check AIPS2

Table continues on the next page...

Table continued from the previous page...

Field	Function
EN_ADD_AIPS2	Enables the AIPS2 address check. 0b - Disable 1b - Enable
24 EN_WR_AIPS2	Enable Write Data Check AIPS2 Enables the write data check for AIPS2. 0b - Disable 1b - Enable
23 EN_ADD_AIPS1	Enable Address Check AIPS1 Enables the address check for AIPS1 path. 0b - Disable 1b - Enable
22 EN_WR_AIPS1	Enable Write Data Check AIPS1 Enables the write data check for AIPS1 path. 0b - Disable 1b - Enable
21 EN_ADD_AIPS0	Enable Address Check AIPS0 Enables the address check for AIPS0 path. 0b - Disable 1b - Enable
20 EN_WR_AIPS0	Enable Write Data Check AIPS0 Enables the write data check for AIPS0 path. 0b - Disable 1b - Enable
19 EN_ADD_QSPI	Enable Address Check QuadSPI Enables the address check for QuadSPI path. 0b - Disable 1b - Enable
18 —	Reserved
17 EN_ADD_TCM	Enable Address Check TCM Enables the address check for TCM 64-bit path.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
16 EN_WR_TCM	Enable Write Data Check TCM Enables the write data check for TCM 64-bit path. 0b - Disable 1b - Enable
15 EN_ADD_PRAM1	Enable Address Check PRAM1 Enables the address check for PRAM1 path. 0b - Disable 1b - Enable
14 EN_WR_PRAM1	Enable Write Data Check PRAM1 Enables the write data check for PRAM1. 0b - Disable 1b - Enable
13 EN_ADD_PRAM0	Enable Address Check PRAM0 Enables the address check for PRAM0 path. 0b - Disable 1b - Enable
12 EN_WR_PRAM0	Enable Write Data Check PRAM0 Enables the write data check for PRAM0 path. 0b - Disable 1b - Enable
11 EN_ADD_PFLASH_PORT2	Enable Address Check P_FLASH_PORT2 Enables the address check for P_FLASH_PORT2 path. 0b - Disable 1b - Enable
10 EN_ADD_PFLASH_PORT1	Enable Address Check P_FLASH_PORT1 Enables the address check for P_FLASH_PORT1 path. 0b - Disable 1b - Enable
9	Enable Address Check P_FLASH_PORT0

Table continues on the next page...

Table continued from the previous page...

Field	Function
EN_ADD_PFLASH_PORT0	Enables the address check for P_FLASH_PORT0 path. 0b - Disable 1b - Enable
8 EN_RD_TCM	Enable Read Data Check TCM Enables the read data check for TCM 32-bit path. 0b - Disable 1b - Enable
7 EN_RD_CM7_1_AHBP	Enable Read Data Check Cortex-M7_1_AHBP Enables the read data check for Cortex-M7_1_AHBP path. 0b - Disable 1b - Enable
6 EN_RD_CM7_1_AHBM	Enable Read Data Check Cortex-M7_1_AHBM Enables the read data check for Cortex-M7_1_AHBM path. 0b - Disable 1b - Enable
5 EN_RD_EMAC	Enable Read Data Check EMAC Enables read data check for EMAC path. 0b - Disable 1b - Enable
4 EN_RD_HSE	Enable Read Data Check HSE Enables the read data check for HSE path. 0b - Disable 1b - Enable
3 —	Reserved
2 EN_RD_EDMA	Enable Read Data Check eDMA Enables the read data check for eDMA path. 0b - Disable 1b - Enable
1	Enable Read Data Check Cortex-M7_0_AHBP Enables the read data check for Cortex-M7_0_AHBP path.

Table continues on the next page...

Table continued from the previous page...

Field	Function
EN_RD_CM7_0_AHBP	0b - Disable 1b - Enable
0 EN_RD_CM7_0_AHBM	Enable Read Data Check Cortex-M7_0_AHBM Enables the read data check for Cortex-M7_0_AHBM path. 0b - Disable 1b - Enable

7.6.3.25 AHB Gasket Configuration (IAHBCFGREG)

Offset

Register	Offset
IAHBCFGREG	700h

Function

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

Access: Privileged mode only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			AIPS2_D...	0			AIPS1_D...	0			QSPI_DI...	0			TCM_DIS...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31-29 —	Reserved
28	AIPS2 Disable Write Optimization

Table continues on the next page...

Table continued from the previous page...

Field	Function
AIPS2_DIS_WR_OPT	Determines whether write burst optimizations in the AIPS2 AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a master. 0b - Enable 1b - Disable
27-25 —	Reserved
24 AIPS1_DIS_WR_OPT	AIPS1 Disable Write Optimization Determines whether write burst optimizations in the AIPS1 AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a master. 0b - Enable 1b - Disable
23-21 —	Reserved
20 QSPI_DIS_WR_OPT	QuadSPI Disable Write Optimization Determines whether write burst optimizations in the QuadSPI AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a master. 0b - Enable 1b - Disable
19-17 —	Reserved
16 TCM_DIS_WR_OPT	TCM Disable Write Optimization Determines whether write burst optimizations in the TCM AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a master. 0b - Enable 1b - Disable
15-13 —	Reserved
12	HSE_B Disable Write Optimization

Table continues on the next page...

Table continued from the previous page...

Field	Function
HSE_DIS_WR_OPT	Determines whether write burst optimizations in the HSE_B AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a master. 0b - Enable 1b - Disable
11-9 —	Reserved
8 DMA_AXBS_S1_DIS_WR_OPT	DMA AXBS S1 Disable Write Optimization Determines whether write burst optimizations in the DMA AXBS S1 AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a master. 0b - Enable 1b - Disable
7-5 —	Reserved
4 DMA_AXBS_S0_DIS_WR_OPT	DMA AXBS S0 Disable Write Optimization Determines whether write burst optimizations in the DMA AXBS S0 AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a master. 0b - Enable 1b - Disable
3-1 —	Reserved
0 EMAC_DIS_WR_OPT	EMAC Disable Write Optimization Determines whether write burst optimizations in the EMAC AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a master. 0b - Enable 1b - Disable

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

Offset

For n = 0 to 239:

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

Function

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

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

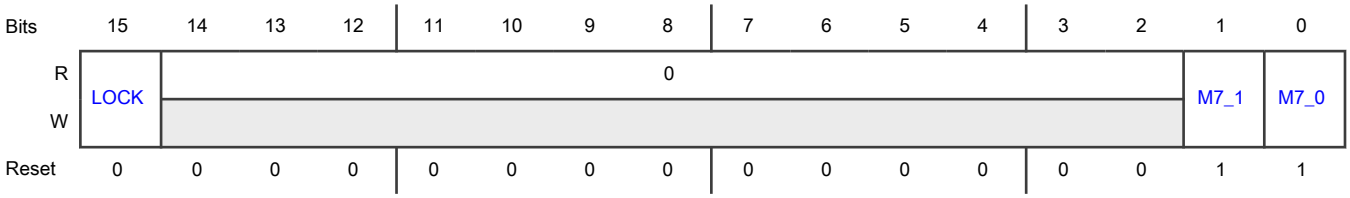
Privileged accesses from noncore (and nondebug) bus masters are treated as RAZ/WI, and any attempted User mode reference terminates with an error. Attempted accesses using a size other than a 16-bit halfword also terminate with an error.

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

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

Access: Privileged mode only

Diagram



Fields

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

Table continues on the next page...

Table continued from the previous page...

Field	Function
M7_0	Enables the corresponding interrupt request to route to Cortex-M7_0. 0b - Routing disabled 1b - Routing enabled

7.7 Glossary

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

Chapter 8
Virtualization Wrapper (VIRT_WRAPPER)

8.1 Chip-specific VIRT_WRAPPER information

8.1.1 System overview of PDAC scheme

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

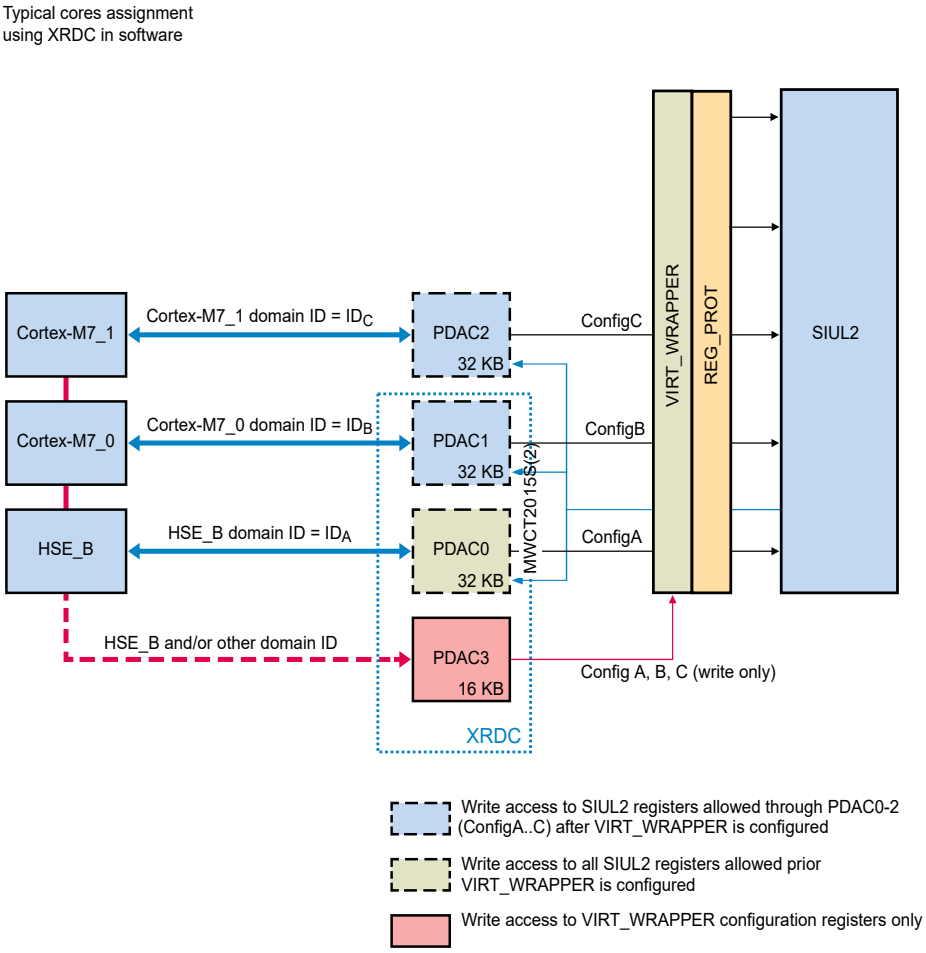


Figure 16. System overview of PDAC scheme

8.1.2 Initial VIRT_WRAPPER operation

Initially, VIRT_WRAPPER:

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

8.1.3 Additional VIRT_WRAPPER details

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

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

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

HSE_B accesses PDAC0 solely for the pads having 11b as the value in their corresponding registers. Also, HSE_B retains the same value for the pads that it stores. In case HSE_B needs to take control of some pins, it can still configure PDAC*n* to be accessible not only from a specific core master, but also from HSE_B, before it allows other cores to execute.

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

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

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

  Then, you can access all these registers only through the assigned PDAC because the VIRT_WRAPPER_REG_C1039_1024 register controls the SIUL2 interrupt registers.
- You cannot access the SIUL2 R/W control registers configured through PDAC3 to be accessible through the ConfigA VIRT_WRAPPER access path, using the ConfigB VIRT_WRAPPER access path.
- You cannot access any of the SIUL2 R/W control registers through ConfigB-C VIRT_WRAPPER access paths prior to VIRT_WRAPPER configuration. By default, any core can access SIUL2 registers through PDAC0 after reset.
- The same PDAC that accesses the main SIUL2 registers is used to access the SIUL2-mirrored registers and soft-lock-bit registers. This is because these registers are a part of the same 16 KB space.
- Access to mirrored SIUL2 registers sets the bit in Soft lock bit as an inherent property of register protection. See the "Register Protection" chapter for details on the soft lock bit behavior, when accessing the mirrored region.
- You cannot access any of the SIUL2 R/W control registers through ConfigB-D VIRT_WRAPPER access paths prior to VIRT_WRAPPER configuration. By default, any core can access SIUL2 registers through PDAC0 after reset.

8.2 Introduction

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

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

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

8.2.1 Overview

Virtualization is implemented by protecting (for example, granting or inhibiting) the access to a register within the virtualized module, dependent on the specific criteria. Virtualized module is the module instance associated with VIRT_WRAPPER. In this chip, SIUL2 is the virtualized module. Registers within virtualized module are virtualized by granting or inhibiting the access to different PDAC slots as encoded within virtualization pad assignments in VIRT_WRAPPER configuration registers. For this purpose, the virtualization information is programmed into VIRT_WRAPPER configuration registers that specify the grouping of registers within the virtualized module. Upon an access to virtualized module through the peripheral interface, the grouping information is exercised to identify whether the corresponding transaction should be granted or inhibited. The following figure depicts the usage of VIRT_WRAPPER in combination with the virtualized module.

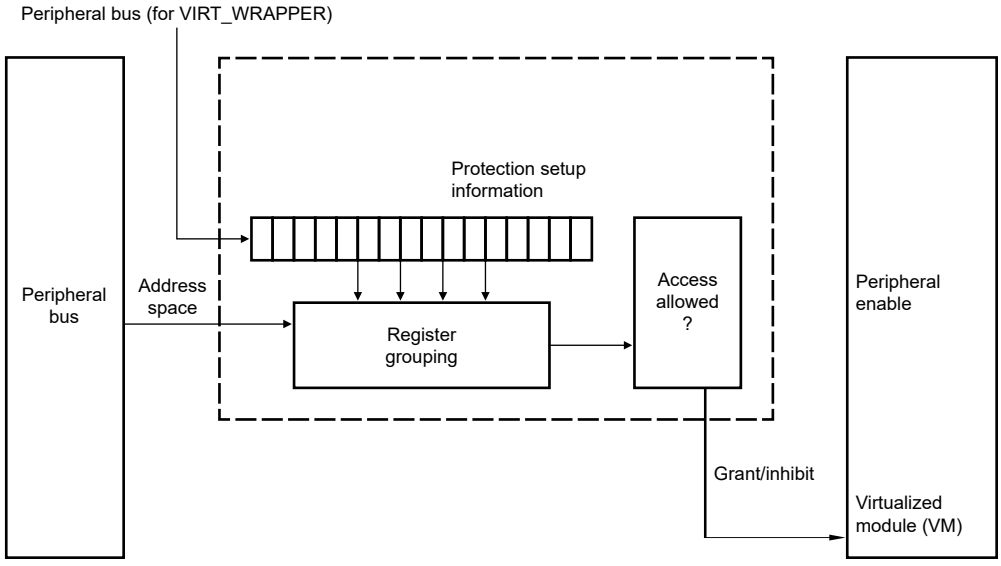


Figure 17. VIRT_WRAPPER block diagram

8.2.2 Features

VIRT_WRAPPER includes these features:

- A programmable chip-specific virtualization setup, capable to select a subset of the available masters in combination with up to four available address spaces

- Virtualizing accesses to the registers within a VM (Virtualized module, such as SIUL2) by inhibiting accesses to a subset of these registers under control of the specified virtualization information. Only write accesses can be inhibited for a specific master or set of masters.

8.2.3 Modes of operation

VIRT_WRAPPER is operable when the VM is operable. For details about the availability of the VM, see the chapter of the corresponding module. When there is no virtualization information specified for the VIRT_WRAPPER module, if XRDC is not configured, PDAC slot 0 is open as the default value of configuration register is 11b that is assigned with PDAC slot 0 and any master can access through it.

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

8.3 Functional description

This section describes the following topics:

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

8.3.1 PDAC-based protection scheme

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

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

Table 32. Protection control information for a single group

Value	Mnemonic	Protection	Description
00	PDAC1	Restricted access of registers, MSCR, IMCR, to PDAC slot 1	Protection setup information for PDAC slot 1 is used.
01	PDAC2	Restricted access of registers, MSCR, IMCR, to PDAC slot 2	Protection setup information for PDAC slot 2 is used.
11	PDAC0	By default, any core can access SIUL2 registers through PDAC slot 0. This process is called ANY_MASTER.	Protection setup information for PDAC slot 0 is used.

8.3.2 Clocking

This module has no major clocking considerations.

8.3.3 Register group mapping for SIUL2

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

8.3.3.1 Virtualization of pad output registers

Any functional pad (controlled by the SIUL2 instance) is assigned to a separate protection group. This scheme excludes the pads that an MSCR register does not control within an SIUL2 instance (for example, power supply pads). As a result, the protection

granularity defined by the virtualization information is a single pad. The control information associated with this protection group affects all the registers related to this pad.

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

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

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

As the protection granularity is a single pad, any virtualization information associated with this pad affects the corresponding MCSR and GPDO registers directly.

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

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

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

8.3.3.2 Virtualization of input multiplexing control registers

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

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

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

As the protection granularity is a single INMUX, any virtualization information associated with this INMUX affects the corresponding IMCR directly.

8.3.3.3 Virtualization of interrupt control registers

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

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

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

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

Table 33. SIUL2 memory map overview and protection

Offset range	Register	Size (bits)	Protected	Description
0000–000Fh	MIDR1, MIDR2	32	N	Read-only registers, not protected
0010–00C3h	Interrupt registers	32	Y	Protected as a single group—no individual protection of related registers
0240–A3Fh [1]	MSCR	32	Y	Amount of registers defines maximum number of PADs to be controlled and protected
0A40–0123Fh	IMCR	32	Y	Amount of registers defines maximum number of INMUXes to be controlled and protected
1300–14FFh	GPDO GPDO[0] (8-bit) register controls single PAD[0] pad means that if PAD[0] is assigned to PDAC slot 1 then PDAC slot 1 can write to GPDO[0] (8-bit).	8	Y	There are fewer GPDO registers than MSCR registers, as some PADs are not made available as GPIO PADs, but their electrical characteristics can still be programmed
1500–16FFh	GPDI	8	N	PAD input register, not protected
1700–173Fh	PGPDO PGPDO[0] (16-bit) register controls PAD[0-15] pads meaning if only all the PADs[0-15] are assigned to PDAC slot 1, then PDAC slot 1 can write to PGPDO[0] (16-bit).	16	Y	Writable with 8-, 16-, and as a pair with 32-bit accesses
1740–177Fh	PGPDI	16	N	PAD input register, not protected
1780–17FFh	MPGPDO MPGPDO[0] (32-bit) register controls PAD[0-15] pads meaning if only all the PADs[0-15] are assigned to PDAC slot 1, then PDAC slot 1 can write to MPGPDO[0] (32-bit).	32	Y	Only writable with 32-bit accesses
2010–20C3h	Reserved	—	—	—
2240–2A3Fh	Reserved	—	—	—
2A40–323Fh	Reserved	—	—	—
3300–34FFh	Reserved	—	—	—

Table continues on the next page...

Table 33. SIUL2 memory map overview and protection (continued)

Offset range	Register	Size (bits)	Protected	Description
4000–47FFh	Reserved	—	—	—
4800–48FFh	Reserved	—	—	—

1. The SIUL2 block guide specifies the support of a maximum of 512 MSCRs.

8.4 External signals

This module has no external signals.

8.5 Initialization

This module does not require initialization.

8.6 Application Information

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

8.7 VIRT_WRAPPER memory map register descriptions

NOTE

Access to reserved spaces outside the register bank and holes (unimplemented registers) within register bank generates the transfer error. Access to offset 104h does not generate any transfer error.

8.7.1 VIRT_WRAPPER memory map

VIRT_WRAPPER base address: 402A_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Parameter_n Register (REG_A15_0)	32	RW	FFFF_FFFFh
4h	Parameter_n Register (REG_A31_16)	32	RW	FFFF_FFFFh
8h	Parameter_n Register (REG_A47_32)	32	RW	FFFF_FFFFh
Ch	Parameter_n Register (REG_A63_48)	32	RW	FFFF_FFFFh
10h	Parameter_n Register (REG_A79_64)	32	RW	FFFF_FFFFh
14h	Parameter_n Register (REG_A95_80)	32	RW	FFFF_FFFFh
18h	Parameter_n Register (REG_A111_96)	32	RW	FFFF_FFFFh
1Ch	Parameter_n Register (REG_A127_112)	32	RW	FFFF_FFFFh
20h	Parameter_n Register (REG_A143_128)	32	RW	FFFF_FFFFh
24h	Parameter_n Register (REG_A159_144)	32	RW	FFFF_FFFFh
28h	Parameter_n Register (REG_A175_160)	32	RW	FFFF_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2Ch	Parameter_n Register (REG_A191_176)	32	RW	FFFF_FFFFh
30h	Parameter_n Register (REG_A207_192)	32	RW	FFFF_FFFFh
34h	Parameter_n Register (REG_A223_208)	32	RW	FFFF_FFFFh
38h	Parameter_n Register (REG_A239_224)	32	RW	FFFF_FFFFh
3Ch	Parameter_n Register (REG_A255_240)	32	RW	FFFF_FFFFh
40h	Parameter_n Register (REG_A271_256)	32	RW	FFFF_FFFFh
44h	Parameter_n Register (REG_A287_272)	32	RW	FFFF_FFFFh
48h	Parameter_n Register (REG_A303_288)	32	RW	FFFF_FFFFh
4Ch	Parameter_n Register (REG_A319_304)	32	RW	FFFF_FFFFh
50h	Parameter_n Register (REG_A335_320)	32	RW	FFFF_FFFFh
54h	Parameter_n Register (REG_A351_336)	32	RW	FFFF_FFFFh
58h	Parameter_n Register (REG_A367_352)	32	RW	FFFF_FFFFh
5Ch	Parameter_n Register (REG_A383_368)	32	RW	FFFF_FFFFh
60h	Parameter_n Register (REG_A399_384)	32	RW	FFFF_FFFFh
64h	Parameter_n Register (REG_A415_400)	32	RW	FFFF_FFFFh
68h	Parameter_n Register (REG_A431_416)	32	RW	FFFF_FFFFh
6Ch	Parameter_n Register (REG_A447_432)	32	RW	FFFF_FFFFh
70h	Parameter_n Register (REG_A463_448)	32	RW	FFFF_FFFFh
74h	Parameter_n Register (REG_A479_464)	32	RW	FFFF_FFFFh
78h	Parameter_n Register (REG_A495_480)	32	RW	FFFF_FFFFh
7Ch	Parameter_n Register (REG_A511_496)	32	RW	FFFF_FFFFh
80h	Parameter_n Register (REG_B527_512)	32	RW	FFFF_FFFFh
84h	Parameter_n Register (REG_B543_528)	32	RW	FFFF_FFFFh
88h	Parameter_n Register (REG_B559_544)	32	RW	FFFF_FFFFh
8Ch	Parameter_n Register (REG_B575_560)	32	RW	FFFF_FFFFh
90h	Parameter_n Register (REG_B591_576)	32	RW	FFFF_FFFFh
94h	Parameter_n Register (REG_B607_592)	32	RW	FFFF_FFFFh
98h	Parameter_n Register (REG_B623_608)	32	RW	FFFF_FFFFh
9Ch	Parameter_n Register (REG_B639_624)	32	RW	FFFF_FFFFh
A0h	Parameter_n Register (REG_B655_640)	32	RW	FFFF_FFFFh
A4h	Parameter_n Register (REG_B671_656)	32	RW	FFFF_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A8h	Parameter_n Register (REG_B687_672)	32	RW	FFFF_FFFFh
ACh	Parameter_n Register (REG_B703_688)	32	RW	FFFF_FFFFh
B0h	Parameter_n Register (REG_B719_704)	32	RW	FFFF_FFFFh
B4h	Parameter_n Register (REG_B735_720)	32	RW	FFFF_FFFFh
B8h	Parameter_n Register (REG_B751_736)	32	RW	FFFF_FFFFh
BCh	Parameter_n Register (REG_B767_752)	32	RW	FFFF_FFFFh
C0h	Parameter_n Register (REG_B783_768)	32	RW	FFFF_FFFFh
C4h	Parameter_n Register (REG_B799_784)	32	RW	FFFF_FFFFh
C8h	Parameter_n Register (REG_B815_800)	32	RW	FFFF_FFFFh
CCh	Parameter_n Register (REG_B831_816)	32	RW	FFFF_FFFFh
D0h	Parameter_n Register (REG_B847_832)	32	RW	FFFF_FFFFh
D4h	Parameter_n Register (REG_B863_848)	32	RW	FFFF_FFFFh
D8h	Parameter_n Register (REG_B879_864)	32	RW	FFFF_FFFFh
DCh	Parameter_n Register (REG_B895_880)	32	RW	FFFF_FFFFh
E0h	Parameter_n Register (REG_B911_896)	32	RW	FFFF_FFFFh
E4h	Parameter_n Register (REG_B927_912)	32	RW	FFFF_FFFFh
E8h	Parameter_n Register (REG_B943_928)	32	RW	FFFF_FFFFh
ECh	Parameter_n Register (REG_B959_944)	32	RW	FFFF_FFFFh
F0h	Parameter_n Register (REG_B975_960)	32	RW	FFFF_FFFFh
F4h	Parameter_n Register (REG_B991_976)	32	RW	FFFF_FFFFh
F8h	Parameter_n Register (REG_B1007_992)	32	RW	FFFF_FFFFh
FCh	Parameter_n Register (REG_B1023_1008)	32	RW	FFFF_FFFFh
100h	Parameter_n Register (REG_C1039_1024)	32	RW	FFFF_FFFFh

8.7.2 Parameter_n Register (REG_A15_0 - REG_A511_496)

Offset

Register	Offset
REG_A15_0	0h
REG_A31_16	4h
REG_A47_32	8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
REG_A63_48	Ch
REG_A79_64	10h
REG_A95_80	14h
REG_A111_96	18h
REG_A127_112	1Ch
REG_A143_128	20h
REG_A159_144	24h
REG_A175_160	28h
REG_A191_176	2Ch
REG_A207_192	30h
REG_A223_208	34h
REG_A239_224	38h
REG_A255_240	3Ch
REG_A271_256	40h
REG_A287_272	44h
REG_A303_288	48h
REG_A319_304	4Ch
REG_A335_320	50h
REG_A351_336	54h
REG_A367_352	58h
REG_A383_368	5Ch
REG_A399_384	60h
REG_A415_400	64h
REG_A431_416	68h
REG_A447_432	6Ch
REG_A463_448	70h
REG_A479_464	74h
REG_A495_480	78h
REG_A511_496	7Ch

Function

This register set is for PAD0-511. They control MSCR, GPDO, PGPDO, and MPGPDO. Two bits assigned per PDAC have attributes of one of the implemented PDACs:

- 00-SIUL2_VIRTWRAPPER_PDAC1

- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-Reserved
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

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

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PAD_15		PAD_14		PAD_13		PAD_12		PAD_11		PAD_10		PAD_9		PAD_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PAD_7		PAD_6		PAD_5		PAD_4		PAD_3		PAD_2		PAD_1		PAD_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 PAD_15	PAD_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 PAD_14	PAD_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 PAD_13	PAD_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 PAD_12	PAD_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 PAD_11	PAD_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 PAD_10	PAD_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 PAD_9	PAD_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 PAD_8	PAD_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 PAD_7	PAD_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 PAD_6	PAD_6 00b - SIUL2_VIRTWRAPPER_PDAC1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 PAD_5	PAD_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 PAD_4	PAD_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 PAD_3	PAD_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 PAD_2	PAD_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 PAD_1	PAD_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 PAD_0	PAD_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - SIUL2_VIRTWRAPPER_PDAC0

8.7.3 Parameter_n Register (REG_B527_512 - REG_B1023_1008)

Offset

Register	Offset
REG_B527_512	80h
REG_B543_528	84h
REG_B559_544	88h
REG_B575_560	8Ch
REG_B591_576	90h
REG_B607_592	94h
REG_B623_608	98h
REG_B639_624	9Ch
REG_B655_640	A0h
REG_B671_656	A4h
REG_B687_672	A8h
REG_B703_688	ACh
REG_B719_704	B0h
REG_B735_720	B4h
REG_B751_736	B8h
REG_B767_752	BCh
REG_B783_768	C0h
REG_B799_784	C4h
REG_B815_800	C8h
REG_B831_816	CCh
REG_B847_832	D0h
REG_B863_848	D4h
REG_B879_864	D8h
REG_B895_880	DCh
REG_B911_896	E0h
REG_B927_912	E4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
REG_B943_928	E8h
REG_B959_944	ECh
REG_B975_960	F0h
REG_B991_976	F4h
REG_B1007_992	F8h
REG_B1023_1008	FCh

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDACs:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-Reserved
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

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

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - Reserved 11b - SIUL2_VIRTWRAPPER_PDAC0

8.7.4 Parameter_n Register (REG_C1039_1024)

Offset

Register	Offset
REG_C1039_1024	100h

Function

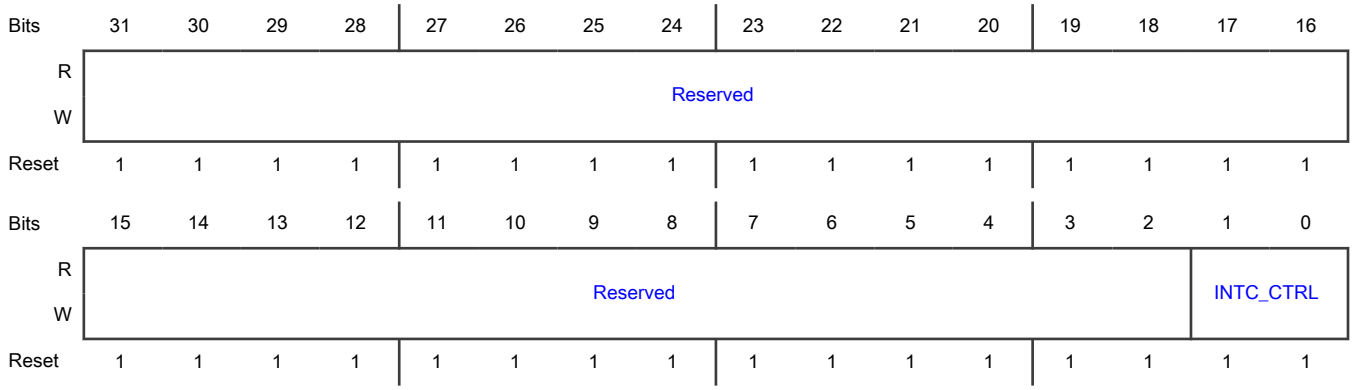
Controls access to DISR0, DIRER0, DIRSR0, IREER0, IFEER0, IFER0, IFMCR, and IFCPR0 interrupt registers. Two bits assigned per PDAC have attributes of one of the implemented PDACs:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-Reserved
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 INTC_CTRL	Interrupt register control This bit controls the interrupt register.

8.8 Glossary

VM Virtualized module, such as SIUL2.

PDAC Peripheral domain access control. PDAC and PDAC slot are interchangeable terms.

Chapter 9
System Integration Unit Lite2 (SIUL2)

9.1 Chip-specific SIUL2 information

9.1.1 Feature availability

This chip:

- Supports input filter enable (MSCR5[IFE]) only for the reset pad (PTA5). For details, see the "Pad signal description" table in the "Signal Multiplexing" chapter.
- Does not implement the open-drain feature. LPI2C directly configures only the pads related to the I2C and LPUART functions in pseudo open-drain when these functions are muxed to the pads. See the LPI2C and LPUART chapters for more information.
- Reserves GPDO[25:24] and PGPDO1[7:6] because PTA24 and PTA25 are input-only pins.

NOTE

EIRQ[0-15] can be used either for interrupt or DMA request. EIRQ[16-31] can only be used for interrupt request.

9.1.2 Mapping of MSCR and IMCR instances

- CR numbers 0-511 correspond to the MSCR instances.
- CR numbers 512-1023 correspond to the IMCR instances. IMCRs defined in the attached IOMUX file have an offset of 512 with respect to the IMCR number defined in SIUL2 memory map section.

NOTE

IMCR register only supports 32 bit access, any other access might result in unexpected data. See IOMUX file attached to this document for the reset value and exact number of IMCR and MSCR register instances.

9.2 Overview

SIUL2 provides control over all electrical pin controls and ports with 16 bits of bidirectional, general-purpose input and output signals. It enables you to select the functions and electrical characteristics that appear on external chip pins, controls the multiplexing of internal signals from one module to another and controls chip I/O, and supports as many as 32 external interrupts with trigger event configuration.

9.2.1 Block diagram

Provides dedicated pad control to general-purpose pads that can be configured as either inputs or outputs. It provides registers for you to read values from GPIO pads configured as inputs and to write values to GPIO pads configured as outputs:

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

You can access GPIO data registers in various ways to allow port access and bit manipulation without read-modify-write operations:

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

You can configure external interrupt sources at the chip level to be used with any chip pad. You can configure interrupt sources to have a digital filter to reject short glitches on the inputs. The external interrupt/DMA requests map to the interrupt request pins (REQ) in the chip packages.

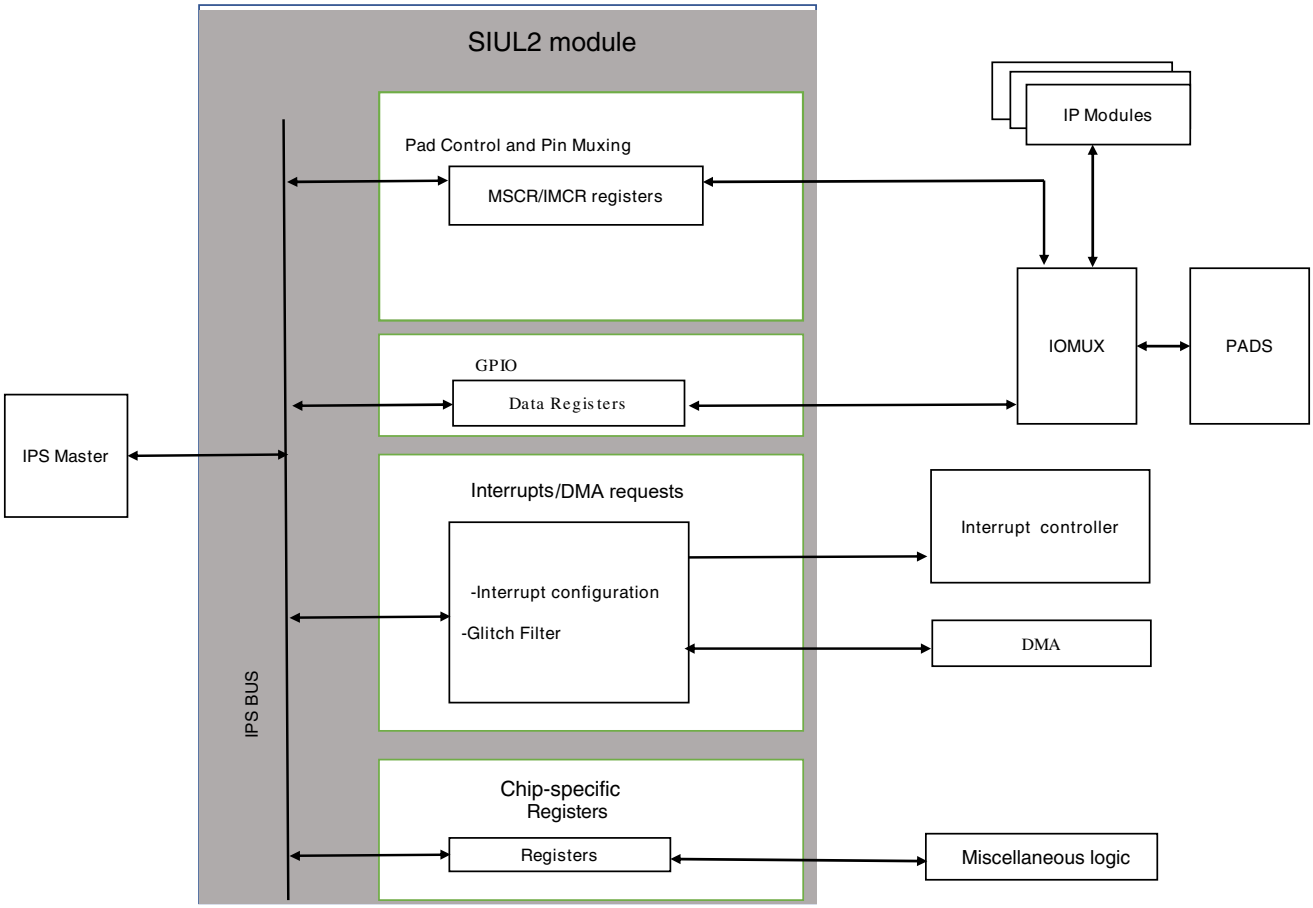


Figure 18. Block diagram

9.2.2 Features

SIUL2 has the following features:

- Supports 1 to 32 GPIO ports with data control:
  - Drives data to as many as 16 independent I/O channels
  - Samples data from as many as 16 independent I/O channels
- Supports read or write of two 16-bit registers with one access for a 32-bit port
- Supports external interrupt/DMA requests:
  - 1 to 32 programmable digital glitch filters, one for each interrupt REQ pin
  - Edge detection
- Supports Multiplexed Signal Configuration Registers (MSCR) to configure the electrical parameters and settings for as many as 512 functional pads.

See the interrupt map file and the DMAMUX map file attached to this document for mapping of interrupt and DMA sources to interrupt vectors and DMA channels.

9.3 Functional description

9.3.1 Pad Control

SIUL2 can control the electrical characteristics of around 512 pads. It provides a consistent interface for all pads, both on a by-port and a by-bit basis.

The setting of each pad out of reset is fixed per chip but you can configure it individually. This way you can select special pull settings or peripheral pad ownership.

You can configure each pad independently of all other pads on the chip or other pads grouped within a single port. Thus, you can group different pad types together in ports and operate the pads individually. Grouping the various functions for each pad into a single register allows you to configure each pad with a single write to a register, which further allows you to duplicate software for similar pads with index changes.

9.3.2 General Purpose Input and Output pads (GPIO)

SIUL2 allows each pad to be configured as either of the following:

- General Purpose Input or Output pad (GPIO)
- A pad for one or more alternate functions (input or output) determined by the peripheral that uses the pad

You can also implement the GPIO pads without any alternate function.

SIUL2 can manage as many as 512 GPIO pads organized as ports that can be accessed for data reads and writes as 8-bit, 16-bit, or 32-bit.

All port accesses are identical, with each read or write performed only at a different location to access a different port width:

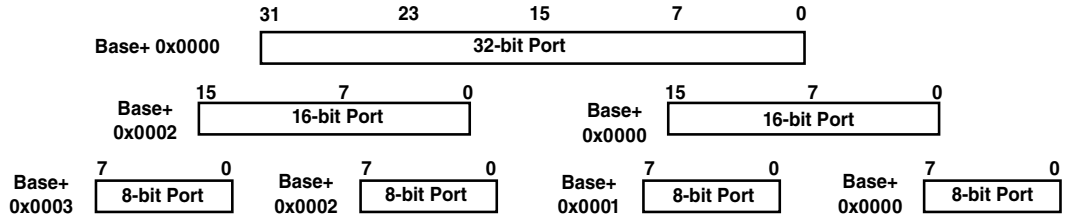


Figure 19. Data Port example arrangement showing configuration for different port width accesses

SIUL2 has separate data input and data output registers for all pads. You can thus directly read back an input or output value of a pad to validate what is actually present on the pad instead of confirming the value that was written to the data input registers.

- The data output registers support both read and write operations.
- The data input registers support read access only.

When a pad is configured to use one of its alternate functions, the data input values reflect the respective value of the pad. If you perform a write operation to the data output register for a pad configured as an alternate function (non-GPIO), this write will not be reflected by the pad value until re-configured to GPIO. All general purpose pads are implemented as bidirectional.

If bidirectional operation impacts performance or is not required for a pad function, you can limit the functionality of the pad to input only.

9.3.3 Clocking

This module has no clocking considerations.

9.3.4 External interrupts

SIUL2 supports one to 32 external interrupts allocated to pads by the chip.

See the interrupt map file and the DMAMUX map file attached to this document (device reference manual) for mapping of interrupt and DMA sources to interrupt vectors and DMA channels.

SIUL2 supports one to four interrupt vectors to the interrupt controller of the chip .Each interrupt vector can support as many as eight external interrupt sources from the chip pads.

All the external interrupt pads within a single group have equal priority. You are responsible for searching through the group of sources in the appropriate way for the application.

NOTE

Glitch filters applied to external interrupts require a running internal oscillator clock. If such a clock is not available, enabling the glitch filter on an external interrupt disables the interrupt.

The external interrupt signals from a pad have internal synchronizers. Therefore, the width of the interrupt signals should be at least 2.5 or 3 times the Internal RC Oscillator (IRC) clock cycles to correctly capture the interrupts.

9.3.4.1 External interrupt initialization

Follow these steps to enable external interrupts:

NOTE

If you do not follow these steps you may get a false interrupt flag during interrupt initialization.

1. Set the appropriate IFER[IFE*n*] fields to enable the glitch filter.
2. Write 0 to DIRER0[EIRE*n*] to mask interrupts.
3. Write 1 to the appropriate IREE*n* fields in the IREER0 register and IFEE*n* fields in the IFEER0 register as needed to select the pin polarity.
4. Configure the appropriate bits in the MSCR register instances for the external interrupt pin(s):
    a. Clear the OBE and ODE bits to disable output.
    b. Set the IBE bit to enable the input buffer of the pin.
    c. If you are using the internal pull-up or pull-down, configure the appropriate PUE and PUS fields.

    NOTE

    When you chose external interrupt inputs for external interrupt pins, do not configure them as outputs (that is, MSCR[OBE] bits must not be to 0) because it can cause false interrupts detection (such as from a GPIO configuration).
5. Write the appropriate DIRSR0[DIRS*n*] bits to select a request between DMA or interrupt.
6. Select the desired glitch filter setup for the pins:
    a. Write the appropriate value to IFMCR*n*[MAXCNT] register for the respective external interrupt to the filter counter.
    b. Write the appropriate value to IFCPR[IFCP] to set the filter clock prescaler.
    c. Set the appropriate IFER0[IFE*n*] bits to enable the glitch filter for the external interrupt pins.
7. Write to the appropriate DISR0[EIF*n*] bits to clear any flags.
8. Write to the appropriate DIRER0[EIRE*n*] bits to enable the interrupt pins.

9.3.4.2 External interrupt management

You can enable or disable each interrupt independently using a single rolled up register, DIRER0.

You can configure a pad defined as an external interrupt to recognize interrupts with an active rising edge, an active falling edge, or both, using IREER and IFEER registers.

NOTE

You must not disable both edge events of a given interrupt.

Each external interrupt has an individual flag, held in DISR0 register. DISR0 is a clear-by-write-1 register, which prevents inadvertent overwriting of other flags in the same register.

This figure provides an overview of the external interrupt implementation:

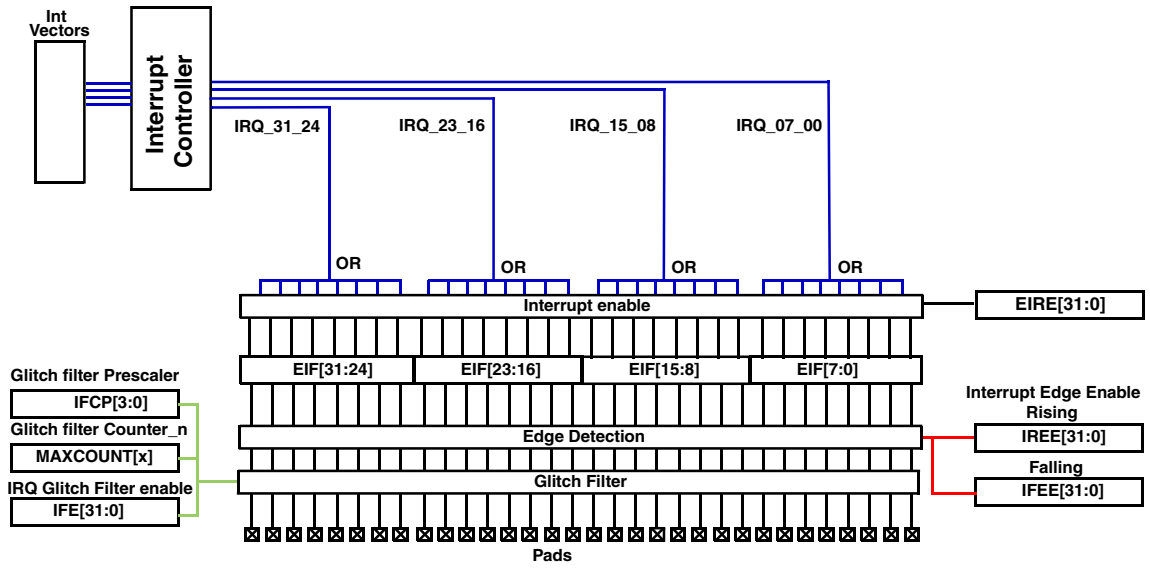


Figure 20. External interrupt pad diagram

9.3.4.3 External interrupt request

The REQ input pins on the chip are the sources for interrupt or DMA requests. The chip provides one possible interrupt vector for SIUL2. The 32 interrupt request sources map to vectors and channels as shown in Table 34.

Table 34. Interrupt source mapping to SIUL2 interrupt request output for 32 interrupt sources

Vector/Channel #	Interrupt vector source
0	REQ[07] | REQ[06] | REQ[05] | REQ[04] | REQ[03] | REQ[02] | REQ[01] | REQ[00]
1	REQ[15] | REQ[14] | REQ[13] | REQ[12] | REQ[11] | REQ[10] | REQ[09] | REQ[08]
2	REQ[23] | REQ[22] | REQ[21] | REQ[20] | REQ[19] | REQ[18] | REQ[17] | REQ[16]
3	REQ[31] | REQ[30] | REQ[29] | REQ[28] | REQ[27] | REQ[26] | REQ[25] | REQ[24]

9.3.5 DMA requests

The REQ pins on the chip map to the independent DMA request channels in the DMA controller. See the DMAMUX map file attached to this document for mapping of DMA sources to DMA channels.

DISR0 and DIRSR0 registers manage the DMA requests in the following way:

- Service a DMA request to clear the DISR0 register flags.
- Write to the appropriate DIRSR0[DIRSn] field to select a request between DMA or interrupt.
- If you select DMA in DIRSR0 and set the corresponding DISR0 flag for it, SIUL2 sends a DMA request signal as an output.

9.4 External signal description

See the IOMUX file attached to this document for more information.

9.5 Initialization

This module does not require initialization.

9.6 SIUL2 register descriptions

This section describes the SIUL2 registers.

NOTE

- Undocumented register spaces in the SIUL2 memory map, including addresses shown as blanks, are Reserved.
  - Reserved registers or spaces are read as 0.
  - Writes to Reserved registers or spaces generate a transfer error.
- Writes to read-only registers generate a transfer error.

NOTE

- For the array of 8-bit registers GPDO*n* and GPDI*n*:
  - An 8-bit access to an unimplemented address (a "hole") within the array region will generate a transfer error.
  - However, if you do a 16-bit or a 32-bit access and if any register instance is implemented within the accessed range, a transfer error will not be generated even if the range includes a hole.
- For the array of 16-bit registers PGPDO*n* and PGPDI*n*:
  - A 16-bit access to an unimplemented address (a "hole") within the array region will generate a transfer error.
  - However, a 32-bit access does not generate a transfer error for a hole irrespective of whether or not the other 16-bit range includes a register instance.

9.6.1 SIUL2 memory map

SIUL2 base address: 4029_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	SIUL2 MCU ID Register #1 (MIDR1)	32	R	See section
8h	SIUL2 MCU ID Register #2 (MIDR2)	32	R	See section
10h	SIUL2 DMA/Interrupt Status Flag 0 (DISR0)	32	RW	0000_0000h
18h	SIUL2 DMA/Interrupt Request Enable 0 (DIRER0)	32	RW	0000_0000h
20h	SIUL2 DMA/Interrupt Request Select 0 (DIRSR0)	32	RW	0000_0000h
28h	SIUL2 Interrupt Rising-Edge Event Enable 0 (IREER0)	32	RW	0000_0000h
30h	SIUL2 Interrupt Falling-Edge Event Enable 0 (IFEER0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
38h	SIUL2 Interrupt Filter Enable 0 (IFER0)	32	RW	0000_0000h
40h - BCh	SIUL2 Interrupt Filter Maximum Counter (IFMCR0 - IFMCR31)	32	RW	0000_0000h
200h	SIUL2 MCU ID Register #3 (MIDR3)	32	R	See section
204h	SIUL2 MCU ID Register #4 (MIDR4)	32	R	See section
240h	SIUL2 Multiplexed Signal Configuration Register (MSCR0)	32	RW	0000_0000h
244h	SIUL2 Multiplexed Signal Configuration Register (MSCR1)	32	RW	0000_0000h
248h	SIUL2 Multiplexed Signal Configuration Register (MSCR2)	32	RW	0000_0000h
24Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR3)	32	RW	0000_0000h
250h	SIUL2 Multiplexed Signal Configuration Register (MSCR4)	32	RW	0008_2827h
254h	SIUL2 Multiplexed Signal Configuration Register (MSCR5)	32	RW	0000_0000h
258h	SIUL2 Multiplexed Signal Configuration Register (MSCR6)	32	RW	0000_0000h
25Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR7)	32	RW	0000_0000h
260h	SIUL2 Multiplexed Signal Configuration Register (MSCR8)	32	RW	0000_0000h
264h	SIUL2 Multiplexed Signal Configuration Register (MSCR9)	32	RW	0000_0000h
268h	SIUL2 Multiplexed Signal Configuration Register (MSCR10)	32	RW	0000_0127h
26Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR11)	32	RW	0000_0000h
270h	SIUL2 Multiplexed Signal Configuration Register (MSCR12)	32	RW	0000_0003h
274h	SIUL2 Multiplexed Signal Configuration Register (MSCR13)	32	RW	0000_0000h
278h	SIUL2 Multiplexed Signal Configuration Register (MSCR14)	32	RW	0000_0000h
27Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR15)	32	RW	0000_0000h
280h	SIUL2 Multiplexed Signal Configuration Register (MSCR16)	32	RW	0000_0000h
284h	SIUL2 Multiplexed Signal Configuration Register (MSCR17)	32	RW	0000_0000h
288h	SIUL2 Multiplexed Signal Configuration Register (MSCR18)	32	RW	0000_0000h
28Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR19)	32	RW	0000_0000h
290h	SIUL2 Multiplexed Signal Configuration Register (MSCR20)	32	RW	0000_0000h
294h	SIUL2 Multiplexed Signal Configuration Register (MSCR21)	32	RW	0000_0000h
298h	SIUL2 Multiplexed Signal Configuration Register (MSCR22)	32	RW	0000_0000h
29Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR23)	32	RW	0000_0000h
2A0h	SIUL2 Multiplexed Signal Configuration Register (MSCR24)	32	RW	0000_0000h
2A4h	SIUL2 Multiplexed Signal Configuration Register (MSCR25)	32	RW	0000_0000h
2A8h	SIUL2 Multiplexed Signal Configuration Register (MSCR26)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2ACh	SIUL2 Multiplexed Signal Configuration Register (MSCR27)	32	RW	0000_0000h
2B0h	SIUL2 Multiplexed Signal Configuration Register (MSCR28)	32	RW	0000_0000h
2B4h	SIUL2 Multiplexed Signal Configuration Register (MSCR29)	32	RW	0000_0000h
2B8h	SIUL2 Multiplexed Signal Configuration Register (MSCR30)	32	RW	0000_0000h
2BCh	SIUL2 Multiplexed Signal Configuration Register (MSCR31)	32	RW	0000_0000h
2C0h	SIUL2 Multiplexed Signal Configuration Register (MSCR32)	32	RW	0000_0000h
2C4h	SIUL2 Multiplexed Signal Configuration Register (MSCR33)	32	RW	0000_0000h
2C8h	SIUL2 Multiplexed Signal Configuration Register (MSCR34)	32	RW	0000_0000h
2CCh	SIUL2 Multiplexed Signal Configuration Register (MSCR35)	32	RW	0000_0000h
2D0h	SIUL2 Multiplexed Signal Configuration Register (MSCR36)	32	RW	0000_0000h
2D4h	SIUL2 Multiplexed Signal Configuration Register (MSCR37)	32	RW	0000_0000h
2E0h	SIUL2 Multiplexed Signal Configuration Register (MSCR40)	32	RW	0000_0000h
2E4h	SIUL2 Multiplexed Signal Configuration Register (MSCR41)	32	RW	0000_0000h
2E8h	SIUL2 Multiplexed Signal Configuration Register (MSCR42)	32	RW	0000_0000h
2ECh	SIUL2 Multiplexed Signal Configuration Register (MSCR43)	32	RW	0000_0000h
2F0h	SIUL2 Multiplexed Signal Configuration Register (MSCR44)	32	RW	0000_0000h
2F4h	SIUL2 Multiplexed Signal Configuration Register (MSCR45)	32	RW	0000_0000h
2F8h	SIUL2 Multiplexed Signal Configuration Register (MSCR46)	32	RW	0000_0000h
2FCh	SIUL2 Multiplexed Signal Configuration Register (MSCR47)	32	RW	0000_0000h
300h	SIUL2 Multiplexed Signal Configuration Register (MSCR48)	32	RW	0000_0000h
304h	SIUL2 Multiplexed Signal Configuration Register (MSCR49)	32	RW	0000_0000h
308h	SIUL2 Multiplexed Signal Configuration Register (MSCR50)	32	RW	0000_0000h
30Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR51)	32	RW	0000_0000h
310h	SIUL2 Multiplexed Signal Configuration Register (MSCR52)	32	RW	0000_0000h
314h	SIUL2 Multiplexed Signal Configuration Register (MSCR53)	32	RW	0000_0000h
318h	SIUL2 Multiplexed Signal Configuration Register (MSCR54)	32	RW	0000_0000h
31Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR55)	32	RW	0000_0000h
320h	SIUL2 Multiplexed Signal Configuration Register (MSCR56)	32	RW	0000_0000h
324h	SIUL2 Multiplexed Signal Configuration Register (MSCR57)	32	RW	0000_0000h
328h	SIUL2 Multiplexed Signal Configuration Register (MSCR58)	32	RW	0000_0000h
32Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR59)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
330h	SIUL2 Multiplexed Signal Configuration Register (MSCR60)	32	RW	0000_0000h
334h	SIUL2 Multiplexed Signal Configuration Register (MSCR61)	32	RW	0000_0000h
338h	SIUL2 Multiplexed Signal Configuration Register (MSCR62)	32	RW	0000_0000h
33Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR63)	32	RW	0000_0000h
340h	SIUL2 Multiplexed Signal Configuration Register (MSCR64)	32	RW	0000_0000h
344h	SIUL2 Multiplexed Signal Configuration Register (MSCR65)	32	RW	0000_0000h
348h	SIUL2 Multiplexed Signal Configuration Register (MSCR66)	32	RW	0000_4000h
34Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR67)	32	RW	0000_4000h
350h	SIUL2 Multiplexed Signal Configuration Register (MSCR68)	32	RW	0008_2000h
354h	SIUL2 Multiplexed Signal Configuration Register (MSCR69)	32	RW	0008_2800h
358h	SIUL2 Multiplexed Signal Configuration Register (MSCR70)	32	RW	0000_0000h
35Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR71)	32	RW	0000_0000h
360h	SIUL2 Multiplexed Signal Configuration Register (MSCR72)	32	RW	0000_0000h
364h	SIUL2 Multiplexed Signal Configuration Register (MSCR73)	32	RW	0000_0000h
368h	SIUL2 Multiplexed Signal Configuration Register (MSCR74)	32	RW	0000_0000h
36Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR75)	32	RW	0000_0000h
370h	SIUL2 Multiplexed Signal Configuration Register (MSCR76)	32	RW	0000_4000h
374h	SIUL2 Multiplexed Signal Configuration Register (MSCR77)	32	RW	0000_0000h
378h	SIUL2 Multiplexed Signal Configuration Register (MSCR78)	32	RW	0000_0000h
37Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR79)	32	RW	0000_0000h
380h	SIUL2 Multiplexed Signal Configuration Register (MSCR80)	32	RW	0000_4000h
384h	SIUL2 Multiplexed Signal Configuration Register (MSCR81)	32	RW	0000_0000h
388h	SIUL2 Multiplexed Signal Configuration Register (MSCR82)	32	RW	0000_0000h
38Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR83)	32	RW	0000_0000h
390h	SIUL2 Multiplexed Signal Configuration Register (MSCR84)	32	RW	0000_0000h
394h	SIUL2 Multiplexed Signal Configuration Register (MSCR85)	32	RW	0000_0000h
398h	SIUL2 Multiplexed Signal Configuration Register (MSCR86)	32	RW	0000_0000h
39Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR87)	32	RW	0000_0000h
3A0h	SIUL2 Multiplexed Signal Configuration Register (MSCR88)	32	RW	0000_0000h
3A4h	SIUL2 Multiplexed Signal Configuration Register (MSCR89)	32	RW	0000_0000h
3A8h	SIUL2 Multiplexed Signal Configuration Register (MSCR90)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3ACh	SIUL2 Multiplexed Signal Configuration Register (MSCR91)	32	RW	0000_0000h
3B0h	SIUL2 Multiplexed Signal Configuration Register (MSCR92)	32	RW	0000_0000h
3B4h	SIUL2 Multiplexed Signal Configuration Register (MSCR93)	32	RW	0000_0000h
3B8h	SIUL2 Multiplexed Signal Configuration Register (MSCR94)	32	RW	0000_0000h
3BCh	SIUL2 Multiplexed Signal Configuration Register (MSCR95)	32	RW	0000_0000h
3C0h	SIUL2 Multiplexed Signal Configuration Register (MSCR96)	32	RW	0000_0000h
3C4h	SIUL2 Multiplexed Signal Configuration Register (MSCR97)	32	RW	0000_0000h
3C8h	SIUL2 Multiplexed Signal Configuration Register (MSCR98)	32	RW	0000_0000h
3CCh	SIUL2 Multiplexed Signal Configuration Register (MSCR99)	32	RW	0000_0000h
3D0h	SIUL2 Multiplexed Signal Configuration Register (MSCR100)	32	RW	0000_0000h
3D4h	SIUL2 Multiplexed Signal Configuration Register (MSCR101)	32	RW	0000_4000h
3D8h	SIUL2 Multiplexed Signal Configuration Register (MSCR102)	32	RW	0000_4000h
3DCh	SIUL2 Multiplexed Signal Configuration Register (MSCR103)	32	RW	0000_4000h
3E0h	SIUL2 Multiplexed Signal Configuration Register (MSCR104)	32	RW	0000_0000h
3E4h	SIUL2 Multiplexed Signal Configuration Register (MSCR105)	32	RW	0000_0000h
3E8h	SIUL2 Multiplexed Signal Configuration Register (MSCR106)	32	RW	0000_4000h
3ECh	SIUL2 Multiplexed Signal Configuration Register (MSCR107)	32	RW	0000_4000h
3F0h	SIUL2 Multiplexed Signal Configuration Register (MSCR108)	32	RW	0000_4000h
3F4h	SIUL2 Multiplexed Signal Configuration Register (MSCR109)	32	RW	0000_0000h
3F8h	SIUL2 Multiplexed Signal Configuration Register (MSCR110)	32	RW	0000_0000h
3FCh	SIUL2 Multiplexed Signal Configuration Register (MSCR111)	32	RW	0000_0000h
400h	SIUL2 Multiplexed Signal Configuration Register (MSCR112)	32	RW	0000_0000h
404h	SIUL2 Multiplexed Signal Configuration Register (MSCR113)	32	RW	0000_0000h
408h	SIUL2 Multiplexed Signal Configuration Register (MSCR114)	32	RW	0000_0000h
40Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR115)	32	RW	0000_0000h
410h	SIUL2 Multiplexed Signal Configuration Register (MSCR116)	32	RW	0000_0000h
414h	SIUL2 Multiplexed Signal Configuration Register (MSCR117)	32	RW	0000_0000h
418h	SIUL2 Multiplexed Signal Configuration Register (MSCR118)	32	RW	0000_0000h
41Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR119)	32	RW	0000_0000h
420h	SIUL2 Multiplexed Signal Configuration Register (MSCR120)	32	RW	0000_0000h
424h	SIUL2 Multiplexed Signal Configuration Register (MSCR121)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
428h	SIUL2 Multiplexed Signal Configuration Register (MSCR122)	32	RW	0000_0000h
42Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR123)	32	RW	0000_0000h
430h	SIUL2 Multiplexed Signal Configuration Register (MSCR124)	32	RW	0000_0000h
434h	SIUL2 Multiplexed Signal Configuration Register (MSCR125)	32	RW	0000_0000h
438h	SIUL2 Multiplexed Signal Configuration Register (MSCR126)	32	RW	0000_0000h
43Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR127)	32	RW	0000_0000h
440h	SIUL2 Multiplexed Signal Configuration Register (MSCR128)	32	RW	0000_0000h
444h	SIUL2 Multiplexed Signal Configuration Register (MSCR129)	32	RW	0000_0000h
448h	SIUL2 Multiplexed Signal Configuration Register (MSCR130)	32	RW	0000_0000h
44Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR131)	32	RW	0000_0000h
450h	SIUL2 Multiplexed Signal Configuration Register (MSCR132)	32	RW	0000_0000h
454h	SIUL2 Multiplexed Signal Configuration Register (MSCR133)	32	RW	0000_0000h
458h	SIUL2 Multiplexed Signal Configuration Register (MSCR134)	32	RW	0000_0000h
45Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR135)	32	RW	0000_0000h
460h	SIUL2 Multiplexed Signal Configuration Register (MSCR136)	32	RW	0000_4000h
464h	SIUL2 Multiplexed Signal Configuration Register (MSCR137)	32	RW	0000_0000h
468h	SIUL2 Multiplexed Signal Configuration Register (MSCR138)	32	RW	0000_0000h
46Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR139)	32	RW	0000_0000h
470h	SIUL2 Multiplexed Signal Configuration Register (MSCR140)	32	RW	0000_0000h
474h	SIUL2 Multiplexed Signal Configuration Register (MSCR141)	32	RW	0000_0000h
478h	SIUL2 Multiplexed Signal Configuration Register (MSCR142)	32	RW	0000_0000h
47Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR143)	32	RW	0000_0000h
480h	SIUL2 Multiplexed Signal Configuration Register (MSCR144)	32	RW	0000_0000h
484h	SIUL2 Multiplexed Signal Configuration Register (MSCR145)	32	RW	0000_0000h
488h	SIUL2 Multiplexed Signal Configuration Register (MSCR146)	32	RW	0000_0000h
48Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR147)	32	RW	0000_0000h
490h	SIUL2 Multiplexed Signal Configuration Register (MSCR148)	32	RW	0000_0000h
494h	SIUL2 Multiplexed Signal Configuration Register (MSCR149)	32	RW	0000_0000h
498h	SIUL2 Multiplexed Signal Configuration Register (MSCR150)	32	RW	0000_0000h
49Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR151)	32	RW	0000_0000h
4A0h	SIUL2 Multiplexed Signal Configuration Register (MSCR152)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4A4h	SIUL2 Multiplexed Signal Configuration Register (MSCR153)	32	RW	0000_0000h
4A8h	SIUL2 Multiplexed Signal Configuration Register (MSCR154)	32	RW	0000_0000h
4ACh	SIUL2 Multiplexed Signal Configuration Register (MSCR155)	32	RW	0000_0000h
4B0h	SIUL2 Multiplexed Signal Configuration Register (MSCR156)	32	RW	0000_0000h
4B4h	SIUL2 Multiplexed Signal Configuration Register (MSCR157)	32	RW	0000_0000h
4B8h	SIUL2 Multiplexed Signal Configuration Register (MSCR158)	32	RW	0000_0000h
4BCh	SIUL2 Multiplexed Signal Configuration Register (MSCR159)	32	RW	0000_0000h
4C0h	SIUL2 Multiplexed Signal Configuration Register (MSCR160)	32	RW	0000_0000h
4C4h	SIUL2 Multiplexed Signal Configuration Register (MSCR161)	32	RW	0000_0000h
4C8h	SIUL2 Multiplexed Signal Configuration Register (MSCR162)	32	RW	0000_0000h
4CCh	SIUL2 Multiplexed Signal Configuration Register (MSCR163)	32	RW	0000_0000h
4D0h	SIUL2 Multiplexed Signal Configuration Register (MSCR164)	32	RW	0000_0000h
4D4h	SIUL2 Multiplexed Signal Configuration Register (MSCR165)	32	RW	0000_0000h
4D8h	SIUL2 Multiplexed Signal Configuration Register (MSCR166)	32	RW	0000_0000h
4DCh	SIUL2 Multiplexed Signal Configuration Register (MSCR167)	32	RW	0000_0000h
4E0h	SIUL2 Multiplexed Signal Configuration Register (MSCR168)	32	RW	0000_0000h
4E4h	SIUL2 Multiplexed Signal Configuration Register (MSCR169)	32	RW	0000_0000h
4E8h	SIUL2 Multiplexed Signal Configuration Register (MSCR170)	32	RW	0000_0000h
4ECh	SIUL2 Multiplexed Signal Configuration Register (MSCR171)	32	RW	0000_0000h
4F0h	SIUL2 Multiplexed Signal Configuration Register (MSCR172)	32	RW	0000_0000h
4F4h	SIUL2 Multiplexed Signal Configuration Register (MSCR173)	32	RW	0000_0000h
4F8h	SIUL2 Multiplexed Signal Configuration Register (MSCR174)	32	RW	0000_0000h
4FCh	SIUL2 Multiplexed Signal Configuration Register (MSCR175)	32	RW	0000_0000h
500h	SIUL2 Multiplexed Signal Configuration Register (MSCR176)	32	RW	0000_0000h
504h	SIUL2 Multiplexed Signal Configuration Register (MSCR177)	32	RW	0000_0000h
508h	SIUL2 Multiplexed Signal Configuration Register (MSCR178)	32	RW	0000_0000h
50Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR179)	32	RW	0000_0000h
510h	SIUL2 Multiplexed Signal Configuration Register (MSCR180)	32	RW	0000_0000h
514h	SIUL2 Multiplexed Signal Configuration Register (MSCR181)	32	RW	0000_0000h
518h	SIUL2 Multiplexed Signal Configuration Register (MSCR182)	32	RW	0000_0000h
51Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR183)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
520h	SIUL2 Multiplexed Signal Configuration Register (MSCR184)	32	RW	0000_0000h
524h	SIUL2 Multiplexed Signal Configuration Register (MSCR185)	32	RW	0000_0000h
528h	SIUL2 Multiplexed Signal Configuration Register (MSCR186)	32	RW	0000_0000h
52Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR187)	32	RW	0000_0000h
530h	SIUL2 Multiplexed Signal Configuration Register (MSCR188)	32	RW	0000_0000h
534h	SIUL2 Multiplexed Signal Configuration Register (MSCR189)	32	RW	0000_0000h
538h	SIUL2 Multiplexed Signal Configuration Register (MSCR190)	32	RW	0000_0000h
53Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR191)	32	RW	0000_0000h
540h	SIUL2 Multiplexed Signal Configuration Register (MSCR192)	32	RW	0000_0000h
544h	SIUL2 Multiplexed Signal Configuration Register (MSCR193)	32	RW	0000_0000h
548h	SIUL2 Multiplexed Signal Configuration Register (MSCR194)	32	RW	0000_0000h
54Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR195)	32	RW	0000_0000h
550h	SIUL2 Multiplexed Signal Configuration Register (MSCR196)	32	RW	0000_0000h
554h	SIUL2 Multiplexed Signal Configuration Register (MSCR197)	32	RW	0000_0000h
558h	SIUL2 Multiplexed Signal Configuration Register (MSCR198)	32	RW	0000_0000h
55Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR199)	32	RW	0000_0000h
560h	SIUL2 Multiplexed Signal Configuration Register (MSCR200)	32	RW	0000_0000h
564h	SIUL2 Multiplexed Signal Configuration Register (MSCR201)	32	RW	0000_0000h
568h	SIUL2 Multiplexed Signal Configuration Register (MSCR202)	32	RW	0000_0000h
56Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR203)	32	RW	0000_0000h
570h	SIUL2 Multiplexed Signal Configuration Register (MSCR204)	32	RW	0000_0000h
574h	SIUL2 Multiplexed Signal Configuration Register (MSCR205)	32	RW	0000_0000h
578h	SIUL2 Multiplexed Signal Configuration Register (MSCR206)	32	RW	0000_0000h
57Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR207)	32	RW	0000_0000h
580h	SIUL2 Multiplexed Signal Configuration Register (MSCR208)	32	RW	0000_0000h
584h	SIUL2 Multiplexed Signal Configuration Register (MSCR209)	32	RW	0000_0000h
588h	SIUL2 Multiplexed Signal Configuration Register (MSCR210)	32	RW	0000_0000h
58Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR211)	32	RW	0000_0000h
590h	SIUL2 Multiplexed Signal Configuration Register (MSCR212)	32	RW	0000_0000h
594h	SIUL2 Multiplexed Signal Configuration Register (MSCR213)	32	RW	0000_0000h
598h	SIUL2 Multiplexed Signal Configuration Register (MSCR214)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
59Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR215)	32	RW	0000_0000h
5A0h	SIUL2 Multiplexed Signal Configuration Register (MSCR216)	32	RW	0000_0000h
5A4h	SIUL2 Multiplexed Signal Configuration Register (MSCR217)	32	RW	0000_0000h
5A8h	SIUL2 Multiplexed Signal Configuration Register (MSCR218)	32	RW	0000_0000h
5ACh	SIUL2 Multiplexed Signal Configuration Register (MSCR219)	32	RW	0000_0000h
A40h	SIUL2 Input Multiplexed Signal Configuration (IMCR0)	32	RW	0000_0000h
A44h	SIUL2 Input Multiplexed Signal Configuration (IMCR1)	32	RW	0000_0000h
A48h	SIUL2 Input Multiplexed Signal Configuration (IMCR2)	32	RW	0000_0000h
A4Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR3)	32	RW	0000_0000h
A50h	SIUL2 Input Multiplexed Signal Configuration (IMCR4)	32	RW	0000_0000h
A54h	SIUL2 Input Multiplexed Signal Configuration (IMCR5)	32	RW	0000_0000h
A80h	SIUL2 Input Multiplexed Signal Configuration (IMCR16)	32	RW	0000_0000h
A84h	SIUL2 Input Multiplexed Signal Configuration (IMCR17)	32	RW	0000_0000h
A88h	SIUL2 Input Multiplexed Signal Configuration (IMCR18)	32	RW	0000_0000h
A8Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR19)	32	RW	0000_0000h
A90h	SIUL2 Input Multiplexed Signal Configuration (IMCR20)	32	RW	0000_0000h
A94h	SIUL2 Input Multiplexed Signal Configuration (IMCR21)	32	RW	0000_0000h
A98h	SIUL2 Input Multiplexed Signal Configuration (IMCR22)	32	RW	0000_0000h
A9Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR23)	32	RW	0000_0000h
AA0h	SIUL2 Input Multiplexed Signal Configuration (IMCR24)	32	RW	0000_0000h
AA4h	SIUL2 Input Multiplexed Signal Configuration (IMCR25)	32	RW	0000_0000h
AA8h	SIUL2 Input Multiplexed Signal Configuration (IMCR26)	32	RW	0000_0000h
AACh	SIUL2 Input Multiplexed Signal Configuration (IMCR27)	32	RW	0000_0000h
AB0h	SIUL2 Input Multiplexed Signal Configuration (IMCR28)	32	RW	0000_0000h
AB4h	SIUL2 Input Multiplexed Signal Configuration (IMCR29)	32	RW	0000_0000h
AB8h	SIUL2 Input Multiplexed Signal Configuration (IMCR30)	32	RW	0000_0000h
ABCh	SIUL2 Input Multiplexed Signal Configuration (IMCR31)	32	RW	0000_0000h
AC0h	SIUL2 Input Multiplexed Signal Configuration (IMCR32)	32	RW	0000_0000h
AC4h	SIUL2 Input Multiplexed Signal Configuration (IMCR33)	32	RW	0000_0000h
AC8h	SIUL2 Input Multiplexed Signal Configuration (IMCR34)	32	RW	0000_0000h
ACCh	SIUL2 Input Multiplexed Signal Configuration (IMCR35)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
AD0h	SIUL2 Input Multiplexed Signal Configuration (IMCR36)	32	RW	0000_0000h
AD4h	SIUL2 Input Multiplexed Signal Configuration (IMCR37)	32	RW	0000_0000h
AD8h	SIUL2 Input Multiplexed Signal Configuration (IMCR38)	32	RW	0000_0000h
ADCh	SIUL2 Input Multiplexed Signal Configuration (IMCR39)	32	RW	0000_0000h
AE0h	SIUL2 Input Multiplexed Signal Configuration (IMCR40)	32	RW	0000_0000h
AE4h	SIUL2 Input Multiplexed Signal Configuration (IMCR41)	32	RW	0000_0000h
AE8h	SIUL2 Input Multiplexed Signal Configuration (IMCR42)	32	RW	0000_0000h
AECh	SIUL2 Input Multiplexed Signal Configuration (IMCR43)	32	RW	0000_0000h
AF0h	SIUL2 Input Multiplexed Signal Configuration (IMCR44)	32	RW	0000_0000h
AF4h	SIUL2 Input Multiplexed Signal Configuration (IMCR45)	32	RW	0000_0000h
AF8h	SIUL2 Input Multiplexed Signal Configuration (IMCR46)	32	RW	0000_0000h
AFCh	SIUL2 Input Multiplexed Signal Configuration (IMCR47)	32	RW	0000_0000h
B00h	SIUL2 Input Multiplexed Signal Configuration (IMCR48)	32	RW	0000_0000h
B04h	SIUL2 Input Multiplexed Signal Configuration (IMCR49)	32	RW	0000_0000h
B08h	SIUL2 Input Multiplexed Signal Configuration (IMCR50)	32	RW	0000_0000h
B0Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR51)	32	RW	0000_0000h
B10h	SIUL2 Input Multiplexed Signal Configuration (IMCR52)	32	RW	0000_0000h
B14h	SIUL2 Input Multiplexed Signal Configuration (IMCR53)	32	RW	0000_0000h
B18h	SIUL2 Input Multiplexed Signal Configuration (IMCR54)	32	RW	0000_0000h
B1Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR55)	32	RW	0000_0000h
B20h	SIUL2 Input Multiplexed Signal Configuration (IMCR56)	32	RW	0000_0000h
B24h	SIUL2 Input Multiplexed Signal Configuration (IMCR57)	32	RW	0000_0000h
B28h	SIUL2 Input Multiplexed Signal Configuration (IMCR58)	32	RW	0000_0000h
B2Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR59)	32	RW	0000_0000h
B30h	SIUL2 Input Multiplexed Signal Configuration (IMCR60)	32	RW	0000_0000h
B34h	SIUL2 Input Multiplexed Signal Configuration (IMCR61)	32	RW	0000_0000h
B38h	SIUL2 Input Multiplexed Signal Configuration (IMCR62)	32	RW	0000_0000h
B3Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR63)	32	RW	0000_0000h
B40h	SIUL2 Input Multiplexed Signal Configuration (IMCR64)	32	RW	0000_0000h
B44h	SIUL2 Input Multiplexed Signal Configuration (IMCR65)	32	RW	0000_0000h
B48h	SIUL2 Input Multiplexed Signal Configuration (IMCR66)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B4Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR67)	32	RW	0000_0000h
B50h	SIUL2 Input Multiplexed Signal Configuration (IMCR68)	32	RW	0000_0000h
B54h	SIUL2 Input Multiplexed Signal Configuration (IMCR69)	32	RW	0000_0000h
B58h	SIUL2 Input Multiplexed Signal Configuration (IMCR70)	32	RW	0000_0000h
B5Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR71)	32	RW	0000_0000h
B80h	SIUL2 Input Multiplexed Signal Configuration (IMCR80)	32	RW	0000_0000h
B84h	SIUL2 Input Multiplexed Signal Configuration (IMCR81)	32	RW	0000_0000h
B88h	SIUL2 Input Multiplexed Signal Configuration (IMCR82)	32	RW	0000_0000h
B8Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR83)	32	RW	0000_0000h
B90h	SIUL2 Input Multiplexed Signal Configuration (IMCR84)	32	RW	0000_0000h
B94h	SIUL2 Input Multiplexed Signal Configuration (IMCR85)	32	RW	0000_0000h
B98h	SIUL2 Input Multiplexed Signal Configuration (IMCR86)	32	RW	0000_0000h
B9Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR87)	32	RW	0000_0000h
BA0h	SIUL2 Input Multiplexed Signal Configuration (IMCR88)	32	RW	0000_0000h
BA4h	SIUL2 Input Multiplexed Signal Configuration (IMCR89)	32	RW	0000_0000h
BA8h	SIUL2 Input Multiplexed Signal Configuration (IMCR90)	32	RW	0000_0000h
BACh	SIUL2 Input Multiplexed Signal Configuration (IMCR91)	32	RW	0000_0000h
BB0h	SIUL2 Input Multiplexed Signal Configuration (IMCR92)	32	RW	0000_0000h
BB4h	SIUL2 Input Multiplexed Signal Configuration (IMCR93)	32	RW	0000_0000h
BB8h	SIUL2 Input Multiplexed Signal Configuration (IMCR94)	32	RW	0000_0000h
BBCh	SIUL2 Input Multiplexed Signal Configuration (IMCR95)	32	RW	0000_0000h
BC0h	SIUL2 Input Multiplexed Signal Configuration (IMCR96)	32	RW	0000_0000h
BC4h	SIUL2 Input Multiplexed Signal Configuration (IMCR97)	32	RW	0000_0000h
BC8h	SIUL2 Input Multiplexed Signal Configuration (IMCR98)	32	RW	0000_0000h
BCCh	SIUL2 Input Multiplexed Signal Configuration (IMCR99)	32	RW	0000_0000h
BD0h	SIUL2 Input Multiplexed Signal Configuration (IMCR100)	32	RW	0000_0000h
BD4h	SIUL2 Input Multiplexed Signal Configuration (IMCR101)	32	RW	0000_0000h
BD8h	SIUL2 Input Multiplexed Signal Configuration (IMCR102)	32	RW	0000_0000h
BDCh	SIUL2 Input Multiplexed Signal Configuration (IMCR103)	32	RW	0000_0000h
C00h	SIUL2 Input Multiplexed Signal Configuration (IMCR112)	32	RW	0000_0000h
C04h	SIUL2 Input Multiplexed Signal Configuration (IMCR113)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
C08h	SIUL2 Input Multiplexed Signal Configuration (IMCR114)	32	RW	0000_0000h
C0Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR115)	32	RW	0000_0000h
C10h	SIUL2 Input Multiplexed Signal Configuration (IMCR116)	32	RW	0000_0000h
C14h	SIUL2 Input Multiplexed Signal Configuration (IMCR117)	32	RW	0000_0000h
C18h	SIUL2 Input Multiplexed Signal Configuration (IMCR118)	32	RW	0000_0000h
C1Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR119)	32	RW	0000_0000h
C20h	SIUL2 Input Multiplexed Signal Configuration (IMCR120)	32	RW	0000_0000h
C24h	SIUL2 Input Multiplexed Signal Configuration (IMCR121)	32	RW	0000_0000h
C28h	SIUL2 Input Multiplexed Signal Configuration (IMCR122)	32	RW	0000_0000h
C2Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR123)	32	RW	0000_0000h
C30h	SIUL2 Input Multiplexed Signal Configuration (IMCR124)	32	RW	0000_0000h
C34h	SIUL2 Input Multiplexed Signal Configuration (IMCR125)	32	RW	0000_0000h
C38h	SIUL2 Input Multiplexed Signal Configuration (IMCR126)	32	RW	0000_0000h
C3Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR127)	32	RW	0000_0000h
C40h	SIUL2 Input Multiplexed Signal Configuration (IMCR128)	32	RW	0000_0000h
C44h	SIUL2 Input Multiplexed Signal Configuration (IMCR129)	32	RW	0000_0000h
C48h	SIUL2 Input Multiplexed Signal Configuration (IMCR130)	32	RW	0000_0000h
C4Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR131)	32	RW	0000_0000h
C50h	SIUL2 Input Multiplexed Signal Configuration (IMCR132)	32	RW	0000_0000h
C54h	SIUL2 Input Multiplexed Signal Configuration (IMCR133)	32	RW	0000_0000h
C58h	SIUL2 Input Multiplexed Signal Configuration (IMCR134)	32	RW	0000_0000h
C5Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR135)	32	RW	0000_0000h
C80h	SIUL2 Input Multiplexed Signal Configuration (IMCR144)	32	RW	0000_0000h
C84h	SIUL2 Input Multiplexed Signal Configuration (IMCR145)	32	RW	0000_0000h
C88h	SIUL2 Input Multiplexed Signal Configuration (IMCR146)	32	RW	0000_0000h
C8Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR147)	32	RW	0000_0000h
C90h	SIUL2 Input Multiplexed Signal Configuration (IMCR148)	32	RW	0000_0000h
C94h	SIUL2 Input Multiplexed Signal Configuration (IMCR149)	32	RW	0000_0000h
CA0h	SIUL2 Input Multiplexed Signal Configuration (IMCR152)	32	RW	0000_0000h
CA4h	SIUL2 Input Multiplexed Signal Configuration (IMCR153)	32	RW	0000_0000h
CA8h	SIUL2 Input Multiplexed Signal Configuration (IMCR154)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
CACh	SIUL2 Input Multiplexed Signal Configuration (IMCR155)	32	RW	0000_0000h
CB0h	SIUL2 Input Multiplexed Signal Configuration (IMCR156)	32	RW	0000_0000h
CB4h	SIUL2 Input Multiplexed Signal Configuration (IMCR157)	32	RW	0000_0000h
CB8h	SIUL2 Input Multiplexed Signal Configuration (IMCR158)	32	RW	0000_0000h
CBCh	SIUL2 Input Multiplexed Signal Configuration (IMCR159)	32	RW	0000_0000h
CC0h	SIUL2 Input Multiplexed Signal Configuration (IMCR160)	32	RW	0000_0000h
CC4h	SIUL2 Input Multiplexed Signal Configuration (IMCR161)	32	RW	0000_0000h
CC8h	SIUL2 Input Multiplexed Signal Configuration (IMCR162)	32	RW	0000_0000h
CCCh	SIUL2 Input Multiplexed Signal Configuration (IMCR163)	32	RW	0000_0000h
CD0h	SIUL2 Input Multiplexed Signal Configuration (IMCR164)	32	RW	0000_0000h
CD4h	SIUL2 Input Multiplexed Signal Configuration (IMCR165)	32	RW	0000_0000h
CD8h	SIUL2 Input Multiplexed Signal Configuration (IMCR166)	32	RW	0000_0000h
CDCh	SIUL2 Input Multiplexed Signal Configuration (IMCR167)	32	RW	0000_0000h
CE0h	SIUL2 Input Multiplexed Signal Configuration (IMCR168)	32	RW	0000_0000h
CE4h	SIUL2 Input Multiplexed Signal Configuration (IMCR169)	32	RW	0000_0000h
CE8h	SIUL2 Input Multiplexed Signal Configuration (IMCR170)	32	RW	0000_0000h
CECh	SIUL2 Input Multiplexed Signal Configuration (IMCR171)	32	RW	0000_0000h
CF0h	SIUL2 Input Multiplexed Signal Configuration (IMCR172)	32	RW	0000_0000h
CF4h	SIUL2 Input Multiplexed Signal Configuration (IMCR173)	32	RW	0000_0000h
CF8h	SIUL2 Input Multiplexed Signal Configuration (IMCR174)	32	RW	0000_0000h
CFCh	SIUL2 Input Multiplexed Signal Configuration (IMCR175)	32	RW	0000_0000h
D00h	SIUL2 Input Multiplexed Signal Configuration (IMCR176)	32	RW	0000_0000h
D04h	SIUL2 Input Multiplexed Signal Configuration (IMCR177)	32	RW	0000_0000h
D08h	SIUL2 Input Multiplexed Signal Configuration (IMCR178)	32	RW	0000_0000h
D0Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR179)	32	RW	0000_0000h
D10h	SIUL2 Input Multiplexed Signal Configuration (IMCR180)	32	RW	0000_0000h
D14h	SIUL2 Input Multiplexed Signal Configuration (IMCR181)	32	RW	0000_0000h
D18h	SIUL2 Input Multiplexed Signal Configuration (IMCR182)	32	RW	0000_0000h
D1Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR183)	32	RW	0000_0000h
D20h	SIUL2 Input Multiplexed Signal Configuration (IMCR184)	32	RW	0000_0000h
D24h	SIUL2 Input Multiplexed Signal Configuration (IMCR185)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
D28h	SIUL2 Input Multiplexed Signal Configuration (IMCR186)	32	RW	0000_0000h
D2Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR187)	32	RW	0000_0000h
D30h	SIUL2 Input Multiplexed Signal Configuration (IMCR188)	32	RW	0000_0000h
D34h	SIUL2 Input Multiplexed Signal Configuration (IMCR189)	32	RW	0000_0000h
D38h	SIUL2 Input Multiplexed Signal Configuration (IMCR190)	32	RW	0000_0000h
D3Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR191)	32	RW	0000_0000h
D40h	SIUL2 Input Multiplexed Signal Configuration (IMCR192)	32	RW	0000_0000h
D44h	SIUL2 Input Multiplexed Signal Configuration (IMCR193)	32	RW	0000_0000h
D48h	SIUL2 Input Multiplexed Signal Configuration (IMCR194)	32	RW	0000_0000h
D4Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR195)	32	RW	0000_0000h
D50h	SIUL2 Input Multiplexed Signal Configuration (IMCR196)	32	RW	0000_0000h
D54h	SIUL2 Input Multiplexed Signal Configuration (IMCR197)	32	RW	0000_0000h
D58h	SIUL2 Input Multiplexed Signal Configuration (IMCR198)	32	RW	0000_0000h
D5Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR199)	32	RW	0000_0000h
D60h	SIUL2 Input Multiplexed Signal Configuration (IMCR200)	32	RW	0000_0000h
D64h	SIUL2 Input Multiplexed Signal Configuration (IMCR201)	32	RW	0000_0000h
D68h	SIUL2 Input Multiplexed Signal Configuration (IMCR202)	32	RW	0000_0000h
D8Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR211)	32	RW	0000_0000h
D90h	SIUL2 Input Multiplexed Signal Configuration (IMCR212)	32	RW	0000_0000h
D94h	SIUL2 Input Multiplexed Signal Configuration (IMCR213)	32	RW	0000_0000h
D98h	SIUL2 Input Multiplexed Signal Configuration (IMCR214)	32	RW	0000_0000h
D9Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR215)	32	RW	0000_0000h
DA0h	SIUL2 Input Multiplexed Signal Configuration (IMCR216)	32	RW	0000_0000h
DA4h	SIUL2 Input Multiplexed Signal Configuration (IMCR217)	32	RW	0000_0000h
DA8h	SIUL2 Input Multiplexed Signal Configuration (IMCR218)	32	RW	0000_0000h
DACh	SIUL2 Input Multiplexed Signal Configuration (IMCR219)	32	RW	0000_0000h
DB0h	SIUL2 Input Multiplexed Signal Configuration (IMCR220)	32	RW	0000_0000h
DB4h	SIUL2 Input Multiplexed Signal Configuration (IMCR221)	32	RW	0000_0000h
DB8h	SIUL2 Input Multiplexed Signal Configuration (IMCR222)	32	RW	0000_0000h
DBCh	SIUL2 Input Multiplexed Signal Configuration (IMCR223)	32	RW	0000_0000h
DC0h	SIUL2 Input Multiplexed Signal Configuration (IMCR224)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
DC4h	SIUL2 Input Multiplexed Signal Configuration (IMCR225)	32	RW	0000_0000h
DC8h	SIUL2 Input Multiplexed Signal Configuration (IMCR226)	32	RW	0000_0000h
DCCh	SIUL2 Input Multiplexed Signal Configuration (IMCR227)	32	RW	0000_0000h
DD0h	SIUL2 Input Multiplexed Signal Configuration (IMCR228)	32	RW	0000_0000h
DD4h	SIUL2 Input Multiplexed Signal Configuration (IMCR229)	32	RW	0000_0000h
DD8h	SIUL2 Input Multiplexed Signal Configuration (IMCR230)	32	RW	0000_0000h
DDCh	SIUL2 Input Multiplexed Signal Configuration (IMCR231)	32	RW	0000_0000h
DE0h	SIUL2 Input Multiplexed Signal Configuration (IMCR232)	32	RW	0000_0000h
DE4h	SIUL2 Input Multiplexed Signal Configuration (IMCR233)	32	RW	0000_0000h
DE8h	SIUL2 Input Multiplexed Signal Configuration (IMCR234)	32	RW	0000_0000h
DECh	SIUL2 Input Multiplexed Signal Configuration (IMCR235)	32	RW	0000_0000h
DF0h	SIUL2 Input Multiplexed Signal Configuration (IMCR236)	32	RW	0000_0000h
DF4h	SIUL2 Input Multiplexed Signal Configuration (IMCR237)	32	RW	0000_0000h
DF8h	SIUL2 Input Multiplexed Signal Configuration (IMCR238)	32	RW	0000_0000h
DFCh	SIUL2 Input Multiplexed Signal Configuration (IMCR239)	32	RW	0000_0000h
E00h	SIUL2 Input Multiplexed Signal Configuration (IMCR240)	32	RW	0000_0000h
E04h	SIUL2 Input Multiplexed Signal Configuration (IMCR241)	32	RW	0000_0000h
E08h	SIUL2 Input Multiplexed Signal Configuration (IMCR242)	32	RW	0000_0000h
E0Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR243)	32	RW	0000_0000h
E10h	SIUL2 Input Multiplexed Signal Configuration (IMCR244)	32	RW	0000_0000h
E14h	SIUL2 Input Multiplexed Signal Configuration (IMCR245)	32	RW	0000_0000h
E18h	SIUL2 Input Multiplexed Signal Configuration (IMCR246)	32	RW	0000_0000h
E1Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR247)	32	RW	0000_0000h
E20h	SIUL2 Input Multiplexed Signal Configuration (IMCR248)	32	RW	0000_0000h
E24h	SIUL2 Input Multiplexed Signal Configuration (IMCR249)	32	RW	0000_0000h
E28h	SIUL2 Input Multiplexed Signal Configuration (IMCR250)	32	RW	0000_0000h
E2Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR251)	32	RW	0000_0000h
E30h	SIUL2 Input Multiplexed Signal Configuration (IMCR252)	32	RW	0000_0000h
E34h	SIUL2 Input Multiplexed Signal Configuration (IMCR253)	32	RW	0000_0000h
E38h	SIUL2 Input Multiplexed Signal Configuration (IMCR254)	32	RW	0000_0000h
E3Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR255)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
E40h	SIUL2 Input Multiplexed Signal Configuration (IMCR256)	32	RW	0000_0000h
E44h	SIUL2 Input Multiplexed Signal Configuration (IMCR257)	32	RW	0000_0000h
E48h	SIUL2 Input Multiplexed Signal Configuration (IMCR258)	32	RW	0000_0000h
E4Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR259)	32	RW	0000_0000h
E50h	SIUL2 Input Multiplexed Signal Configuration (IMCR260)	32	RW	0000_0000h
E54h	SIUL2 Input Multiplexed Signal Configuration (IMCR261)	32	RW	0000_0000h
E58h	SIUL2 Input Multiplexed Signal Configuration (IMCR262)	32	RW	0000_0000h
E5Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR263)	32	RW	0000_0000h
E60h	SIUL2 Input Multiplexed Signal Configuration (IMCR264)	32	RW	0000_0000h
E64h	SIUL2 Input Multiplexed Signal Configuration (IMCR265)	32	RW	0000_0000h
E68h	SIUL2 Input Multiplexed Signal Configuration (IMCR266)	32	RW	0000_0000h
E6Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR267)	32	RW	0000_0000h
E70h	SIUL2 Input Multiplexed Signal Configuration (IMCR268)	32	RW	0000_0000h
EC4h	SIUL2 Input Multiplexed Signal Configuration (IMCR289)	32	RW	0000_0000h
EC8h	SIUL2 Input Multiplexed Signal Configuration (IMCR290)	32	RW	0000_0000h
ECCh	SIUL2 Input Multiplexed Signal Configuration (IMCR291)	32	RW	0000_0000h
ED0h	SIUL2 Input Multiplexed Signal Configuration (IMCR292)	32	RW	0000_0000h
ED4h	SIUL2 Input Multiplexed Signal Configuration (IMCR293)	32	RW	0000_0000h
ED8h	SIUL2 Input Multiplexed Signal Configuration (IMCR294)	32	RW	0000_0000h
EDCh	SIUL2 Input Multiplexed Signal Configuration (IMCR295)	32	RW	0000_0000h
EE0h	SIUL2 Input Multiplexed Signal Configuration (IMCR296)	32	RW	0000_0000h
EE4h	SIUL2 Input Multiplexed Signal Configuration (IMCR297)	32	RW	0000_0000h
EE8h	SIUL2 Input Multiplexed Signal Configuration (IMCR298)	32	RW	0000_0000h
EECh	SIUL2 Input Multiplexed Signal Configuration (IMCR299)	32	RW	0000_0000h
EF0h	SIUL2 Input Multiplexed Signal Configuration (IMCR300)	32	RW	0000_0000h
EF4h	SIUL2 Input Multiplexed Signal Configuration (IMCR301)	32	RW	0000_0000h
EF8h	SIUL2 Input Multiplexed Signal Configuration (IMCR302)	32	RW	0000_0000h
EFCh	SIUL2 Input Multiplexed Signal Configuration (IMCR303)	32	RW	0000_0000h
F00h	SIUL2 Input Multiplexed Signal Configuration (IMCR304)	32	RW	0000_0000h
F04h	SIUL2 Input Multiplexed Signal Configuration (IMCR305)	32	RW	0000_0000h
F08h	SIUL2 Input Multiplexed Signal Configuration (IMCR306)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
F0Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR307)	32	RW	0000_0000h
F10h	SIUL2 Input Multiplexed Signal Configuration (IMCR308)	32	RW	0000_0000h
F14h	SIUL2 Input Multiplexed Signal Configuration (IMCR309)	32	RW	0000_0000h
F2Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR315)	32	RW	0000_0000h
F30h	SIUL2 Input Multiplexed Signal Configuration (IMCR316)	32	RW	0000_0000h
F34h	SIUL2 Input Multiplexed Signal Configuration (IMCR317)	32	RW	0000_0000h
F38h	SIUL2 Input Multiplexed Signal Configuration (IMCR318)	32	RW	0000_0000h
F3Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR319)	32	RW	0000_0000h
F40h	SIUL2 Input Multiplexed Signal Configuration (IMCR320)	32	RW	0000_0000h
F44h	SIUL2 Input Multiplexed Signal Configuration (IMCR321)	32	RW	0000_0000h
F48h	SIUL2 Input Multiplexed Signal Configuration (IMCR322)	32	RW	0000_0000h
F4Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR323)	32	RW	0000_0000h
F50h	SIUL2 Input Multiplexed Signal Configuration (IMCR324)	32	RW	0000_0000h
F54h	SIUL2 Input Multiplexed Signal Configuration (IMCR325)	32	RW	0000_0000h
F9Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR343)	32	RW	0000_0000h
FA0h	SIUL2 Input Multiplexed Signal Configuration (IMCR344)	32	RW	0000_0000h
FA4h	SIUL2 Input Multiplexed Signal Configuration (IMCR345)	32	RW	0000_0000h
FA8h	SIUL2 Input Multiplexed Signal Configuration (IMCR346)	32	RW	0000_0000h
FACh	SIUL2 Input Multiplexed Signal Configuration (IMCR347)	32	RW	0000_0000h
FB0h	SIUL2 Input Multiplexed Signal Configuration (IMCR348)	32	RW	0000_0000h
FB4h	SIUL2 Input Multiplexed Signal Configuration (IMCR349)	32	RW	0000_0000h
FB8h	SIUL2 Input Multiplexed Signal Configuration (IMCR350)	32	RW	0000_0000h
FBCh	SIUL2 Input Multiplexed Signal Configuration (IMCR351)	32	RW	0000_0000h
FC0h	SIUL2 Input Multiplexed Signal Configuration (IMCR352)	32	RW	0000_0000h
FC4h	SIUL2 Input Multiplexed Signal Configuration (IMCR353)	32	RW	0000_0000h
FC8h	SIUL2 Input Multiplexed Signal Configuration (IMCR354)	32	RW	0000_0000h
FCCh	SIUL2 Input Multiplexed Signal Configuration (IMCR355)	32	RW	0000_0000h
FD0h	SIUL2 Input Multiplexed Signal Configuration (IMCR356)	32	RW	0000_0000h
FD4h	SIUL2 Input Multiplexed Signal Configuration (IMCR357)	32	RW	0000_0000h
FD8h	SIUL2 Input Multiplexed Signal Configuration (IMCR358)	32	RW	0000_0000h
FDCh	SIUL2 Input Multiplexed Signal Configuration (IMCR359)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
FE0h	SIUL2 Input Multiplexed Signal Configuration (IMCR360)	32	RW	0000_0000h
FE4h	SIUL2 Input Multiplexed Signal Configuration (IMCR361)	32	RW	0000_0000h
FE8h	SIUL2 Input Multiplexed Signal Configuration (IMCR362)	32	RW	0000_0000h
FECh	SIUL2 Input Multiplexed Signal Configuration (IMCR363)	32	RW	0000_0000h
FF0h	SIUL2 Input Multiplexed Signal Configuration (IMCR364)	32	RW	0000_0000h
FF4h	SIUL2 Input Multiplexed Signal Configuration (IMCR365)	32	RW	0000_0000h
FF8h	SIUL2 Input Multiplexed Signal Configuration (IMCR366)	32	RW	0000_0000h
FFCh	SIUL2 Input Multiplexed Signal Configuration (IMCR367)	32	RW	0000_0000h
1000h	SIUL2 Input Multiplexed Signal Configuration (IMCR368)	32	RW	0000_0000h
1004h	SIUL2 Input Multiplexed Signal Configuration (IMCR369)	32	RW	0000_0000h
1008h	SIUL2 Input Multiplexed Signal Configuration (IMCR370)	32	RW	0000_0000h
100Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR371)	32	RW	0000_0000h
1010h	SIUL2 Input Multiplexed Signal Configuration (IMCR372)	32	RW	0000_0000h
1014h	SIUL2 Input Multiplexed Signal Configuration (IMCR373)	32	RW	0000_0000h
1018h	SIUL2 Input Multiplexed Signal Configuration (IMCR374)	32	RW	0000_0000h
101Ch	SIUL2 Input Multiplexed Signal Configuration (IMCR375)	32	RW	0000_0000h
1020h	SIUL2 Input Multiplexed Signal Configuration (IMCR376)	32	RW	0000_0000h
1024h	SIUL2 Input Multiplexed Signal Configuration (IMCR377)	32	RW	0000_0000h
1028h	SIUL2 Input Multiplexed Signal Configuration (IMCR378)	32	RW	0000_0000h
1300h	SIUL2 GPIO Pad Data Output (GPDO3)	8	RW	00h
1301h	SIUL2 GPIO Pad Data Output (GPDO2)	8	RW	00h
1302h	SIUL2 GPIO Pad Data Output (GPDO1)	8	RW	00h
1303h	SIUL2 GPIO Pad Data Output (GPDO0)	8	RW	00h
1304h	SIUL2 GPIO Pad Data Output (GPDO7)	8	RW	00h
1305h	SIUL2 GPIO Pad Data Output (GPDO6)	8	RW	00h
1306h	SIUL2 GPIO Pad Data Output (GPDO5)	8	RW	00h
1307h	SIUL2 GPIO Pad Data Output (GPDO4)	8	RW	00h
1308h	SIUL2 GPIO Pad Data Output (GPDO11)	8	RW	00h
1309h	SIUL2 GPIO Pad Data Output (GPDO10)	8	RW	00h
130Ah	SIUL2 GPIO Pad Data Output (GPDO9)	8	RW	00h
130Bh	SIUL2 GPIO Pad Data Output (GPDO8)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
130Ch	SIUL2 GPIO Pad Data Output (GPDO15)	8	RW	00h
130Dh	SIUL2 GPIO Pad Data Output (GPDO14)	8	RW	00h
130Eh	SIUL2 GPIO Pad Data Output (GPDO13)	8	RW	00h
130Fh	SIUL2 GPIO Pad Data Output (GPDO12)	8	RW	00h
1310h	SIUL2 GPIO Pad Data Output (GPDO19)	8	RW	00h
1311h	SIUL2 GPIO Pad Data Output (GPDO18)	8	RW	00h
1312h	SIUL2 GPIO Pad Data Output (GPDO17)	8	RW	00h
1313h	SIUL2 GPIO Pad Data Output (GPDO16)	8	RW	00h
1314h	SIUL2 GPIO Pad Data Output (GPDO23)	8	RW	00h
1315h	SIUL2 GPIO Pad Data Output (GPDO22)	8	RW	00h
1316h	SIUL2 GPIO Pad Data Output (GPDO21)	8	RW	00h
1317h	SIUL2 GPIO Pad Data Output (GPDO20)	8	RW	00h
1318h	SIUL2 GPIO Pad Data Output (GPDO27)	8	RW	00h
1319h	SIUL2 GPIO Pad Data Output (GPDO26)	8	RW	00h
131Ah	SIUL2 GPIO Pad Data Output (GPDO25)	8	RW	00h
131Bh	SIUL2 GPIO Pad Data Output (GPDO24)	8	RW	00h
131Ch	SIUL2 GPIO Pad Data Output (GPDO31)	8	RW	00h
131Dh	SIUL2 GPIO Pad Data Output (GPDO30)	8	RW	00h
131Eh	SIUL2 GPIO Pad Data Output (GPDO29)	8	RW	00h
131Fh	SIUL2 GPIO Pad Data Output (GPDO28)	8	RW	00h
1320h	SIUL2 GPIO Pad Data Output (GPDO35)	8	RW	00h
1321h	SIUL2 GPIO Pad Data Output (GPDO34)	8	RW	00h
1322h	SIUL2 GPIO Pad Data Output (GPDO33)	8	RW	00h
1323h	SIUL2 GPIO Pad Data Output (GPDO32)	8	RW	00h
1326h	SIUL2 GPIO Pad Data Output (GPDO37)	8	RW	00h
1327h	SIUL2 GPIO Pad Data Output (GPDO36)	8	RW	00h
1328h	SIUL2 GPIO Pad Data Output (GPDO43)	8	RW	00h
1329h	SIUL2 GPIO Pad Data Output (GPDO42)	8	RW	00h
132Ah	SIUL2 GPIO Pad Data Output (GPDO41)	8	RW	00h
132Bh	SIUL2 GPIO Pad Data Output (GPDO40)	8	RW	00h
132Ch	SIUL2 GPIO Pad Data Output (GPDO47)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
132Dh	SIUL2 GPIO Pad Data Output (GPDO46)	8	RW	00h
132Eh	SIUL2 GPIO Pad Data Output (GPDO45)	8	RW	00h
132Fh	SIUL2 GPIO Pad Data Output (GPDO44)	8	RW	00h
1330h	SIUL2 GPIO Pad Data Output (GPDO51)	8	RW	00h
1331h	SIUL2 GPIO Pad Data Output (GPDO50)	8	RW	00h
1332h	SIUL2 GPIO Pad Data Output (GPDO49)	8	RW	00h
1333h	SIUL2 GPIO Pad Data Output (GPDO48)	8	RW	00h
1334h	SIUL2 GPIO Pad Data Output (GPDO55)	8	RW	00h
1335h	SIUL2 GPIO Pad Data Output (GPDO54)	8	RW	00h
1336h	SIUL2 GPIO Pad Data Output (GPDO53)	8	RW	00h
1337h	SIUL2 GPIO Pad Data Output (GPDO52)	8	RW	00h
1338h	SIUL2 GPIO Pad Data Output (GPDO59)	8	RW	00h
1339h	SIUL2 GPIO Pad Data Output (GPDO58)	8	RW	00h
133Ah	SIUL2 GPIO Pad Data Output (GPDO57)	8	RW	00h
133Bh	SIUL2 GPIO Pad Data Output (GPDO56)	8	RW	00h
133Ch	SIUL2 GPIO Pad Data Output (GPDO63)	8	RW	00h
133Dh	SIUL2 GPIO Pad Data Output (GPDO62)	8	RW	00h
133Eh	SIUL2 GPIO Pad Data Output (GPDO61)	8	RW	00h
133Fh	SIUL2 GPIO Pad Data Output (GPDO60)	8	RW	00h
1340h	SIUL2 GPIO Pad Data Output (GPDO67)	8	RW	00h
1341h	SIUL2 GPIO Pad Data Output (GPDO66)	8	RW	00h
1342h	SIUL2 GPIO Pad Data Output (GPDO65)	8	RW	00h
1343h	SIUL2 GPIO Pad Data Output (GPDO64)	8	RW	00h
1344h	SIUL2 GPIO Pad Data Output (GPDO71)	8	RW	00h
1345h	SIUL2 GPIO Pad Data Output (GPDO70)	8	RW	00h
1346h	SIUL2 GPIO Pad Data Output (GPDO69)	8	RW	00h
1347h	SIUL2 GPIO Pad Data Output (GPDO68)	8	RW	00h
1348h	SIUL2 GPIO Pad Data Output (GPDO75)	8	RW	00h
1349h	SIUL2 GPIO Pad Data Output (GPDO74)	8	RW	00h
134Ah	SIUL2 GPIO Pad Data Output (GPDO73)	8	RW	00h
134Bh	SIUL2 GPIO Pad Data Output (GPDO72)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
134Ch	SIUL2 GPIO Pad Data Output (GPDO79)	8	RW	00h
134Dh	SIUL2 GPIO Pad Data Output (GPDO78)	8	RW	00h
134Eh	SIUL2 GPIO Pad Data Output (GPDO77)	8	RW	00h
134Fh	SIUL2 GPIO Pad Data Output (GPDO76)	8	RW	00h
1350h	SIUL2 GPIO Pad Data Output (GPDO83)	8	RW	00h
1351h	SIUL2 GPIO Pad Data Output (GPDO82)	8	RW	00h
1352h	SIUL2 GPIO Pad Data Output (GPDO81)	8	RW	00h
1353h	SIUL2 GPIO Pad Data Output (GPDO80)	8	RW	00h
1354h	SIUL2 GPIO Pad Data Output (GPDO87)	8	RW	00h
1355h	SIUL2 GPIO Pad Data Output (GPDO86)	8	RW	00h
1356h	SIUL2 GPIO Pad Data Output (GPDO85)	8	RW	00h
1357h	SIUL2 GPIO Pad Data Output (GPDO84)	8	RW	00h
1358h	SIUL2 GPIO Pad Data Output (GPDO91)	8	RW	00h
1359h	SIUL2 GPIO Pad Data Output (GPDO90)	8	RW	00h
135Ah	SIUL2 GPIO Pad Data Output (GPDO89)	8	RW	00h
135Bh	SIUL2 GPIO Pad Data Output (GPDO88)	8	RW	00h
135Ch	SIUL2 GPIO Pad Data Output (GPDO95)	8	RW	00h
135Dh	SIUL2 GPIO Pad Data Output (GPDO94)	8	RW	00h
135Eh	SIUL2 GPIO Pad Data Output (GPDO93)	8	RW	00h
135Fh	SIUL2 GPIO Pad Data Output (GPDO92)	8	RW	00h
1360h	SIUL2 GPIO Pad Data Output (GPDO99)	8	RW	00h
1361h	SIUL2 GPIO Pad Data Output (GPDO98)	8	RW	00h
1362h	SIUL2 GPIO Pad Data Output (GPDO97)	8	RW	00h
1363h	SIUL2 GPIO Pad Data Output (GPDO96)	8	RW	00h
1364h	SIUL2 GPIO Pad Data Output (GPDO103)	8	RW	00h
1365h	SIUL2 GPIO Pad Data Output (GPDO102)	8	RW	00h
1366h	SIUL2 GPIO Pad Data Output (GPDO101)	8	RW	00h
1367h	SIUL2 GPIO Pad Data Output (GPDO100)	8	RW	00h
1368h	SIUL2 GPIO Pad Data Output (GPDO107)	8	RW	00h
1369h	SIUL2 GPIO Pad Data Output (GPDO106)	8	RW	00h
136Ah	SIUL2 GPIO Pad Data Output (GPDO105)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
136Bh	SIUL2 GPIO Pad Data Output (GPDO104)	8	RW	00h
136Ch	SIUL2 GPIO Pad Data Output (GPDO111)	8	RW	00h
136Dh	SIUL2 GPIO Pad Data Output (GPDO110)	8	RW	00h
136Eh	SIUL2 GPIO Pad Data Output (GPDO109)	8	RW	00h
136Fh	SIUL2 GPIO Pad Data Output (GPDO108)	8	RW	00h
1370h	SIUL2 GPIO Pad Data Output (GPDO115)	8	RW	00h
1371h	SIUL2 GPIO Pad Data Output (GPDO114)	8	RW	00h
1372h	SIUL2 GPIO Pad Data Output (GPDO113)	8	RW	00h
1373h	SIUL2 GPIO Pad Data Output (GPDO112)	8	RW	00h
1374h	SIUL2 GPIO Pad Data Output (GPDO119)	8	RW	00h
1375h	SIUL2 GPIO Pad Data Output (GPDO118)	8	RW	00h
1376h	SIUL2 GPIO Pad Data Output (GPDO117)	8	RW	00h
1377h	SIUL2 GPIO Pad Data Output (GPDO116)	8	RW	00h
1378h	SIUL2 GPIO Pad Data Output (GPDO123)	8	RW	00h
1379h	SIUL2 GPIO Pad Data Output (GPDO122)	8	RW	00h
137Ah	SIUL2 GPIO Pad Data Output (GPDO121)	8	RW	00h
137Bh	SIUL2 GPIO Pad Data Output (GPDO120)	8	RW	00h
137Ch	SIUL2 GPIO Pad Data Output (GPDO127)	8	RW	00h
137Dh	SIUL2 GPIO Pad Data Output (GPDO126)	8	RW	00h
137Eh	SIUL2 GPIO Pad Data Output (GPDO125)	8	RW	00h
137Fh	SIUL2 GPIO Pad Data Output (GPDO124)	8	RW	00h
1380h	SIUL2 GPIO Pad Data Output (GPDO131)	8	RW	00h
1381h	SIUL2 GPIO Pad Data Output (GPDO130)	8	RW	00h
1382h	SIUL2 GPIO Pad Data Output (GPDO129)	8	RW	00h
1383h	SIUL2 GPIO Pad Data Output (GPDO128)	8	RW	00h
1384h	SIUL2 GPIO Pad Data Output (GPDO135)	8	RW	00h
1385h	SIUL2 GPIO Pad Data Output (GPDO134)	8	RW	00h
1386h	SIUL2 GPIO Pad Data Output (GPDO133)	8	RW	00h
1387h	SIUL2 GPIO Pad Data Output (GPDO132)	8	RW	00h
1388h	SIUL2 GPIO Pad Data Output (GPDO139)	8	RW	00h
1389h	SIUL2 GPIO Pad Data Output (GPDO138)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
138Ah	SIUL2 GPIO Pad Data Output (GPDO137)	8	RW	00h
138Bh	SIUL2 GPIO Pad Data Output (GPDO136)	8	RW	00h
138Ch	SIUL2 GPIO Pad Data Output (GPDO143)	8	RW	00h
138Dh	SIUL2 GPIO Pad Data Output (GPDO142)	8	RW	00h
138Eh	SIUL2 GPIO Pad Data Output (GPDO141)	8	RW	00h
138Fh	SIUL2 GPIO Pad Data Output (GPDO140)	8	RW	00h
1390h	SIUL2 GPIO Pad Data Output (GPDO147)	8	RW	00h
1391h	SIUL2 GPIO Pad Data Output (GPDO146)	8	RW	00h
1392h	SIUL2 GPIO Pad Data Output (GPDO145)	8	RW	00h
1393h	SIUL2 GPIO Pad Data Output (GPDO144)	8	RW	00h
1394h	SIUL2 GPIO Pad Data Output (GPDO151)	8	RW	00h
1395h	SIUL2 GPIO Pad Data Output (GPDO150)	8	RW	00h
1396h	SIUL2 GPIO Pad Data Output (GPDO149)	8	RW	00h
1397h	SIUL2 GPIO Pad Data Output (GPDO148)	8	RW	00h
1398h	SIUL2 GPIO Pad Data Output (GPDO155)	8	RW	00h
1399h	SIUL2 GPIO Pad Data Output (GPDO154)	8	RW	00h
139Ah	SIUL2 GPIO Pad Data Output (GPDO153)	8	RW	00h
139Bh	SIUL2 GPIO Pad Data Output (GPDO152)	8	RW	00h
139Ch	SIUL2 GPIO Pad Data Output (GPDO159)	8	RW	00h
139Dh	SIUL2 GPIO Pad Data Output (GPDO158)	8	RW	00h
139Eh	SIUL2 GPIO Pad Data Output (GPDO157)	8	RW	00h
139Fh	SIUL2 GPIO Pad Data Output (GPDO156)	8	RW	00h
13A0h	SIUL2 GPIO Pad Data Output (GPDO163)	8	RW	00h
13A1h	SIUL2 GPIO Pad Data Output (GPDO162)	8	RW	00h
13A2h	SIUL2 GPIO Pad Data Output (GPDO161)	8	RW	00h
13A3h	SIUL2 GPIO Pad Data Output (GPDO160)	8	RW	00h
13A4h	SIUL2 GPIO Pad Data Output (GPDO167)	8	RW	00h
13A5h	SIUL2 GPIO Pad Data Output (GPDO166)	8	RW	00h
13A6h	SIUL2 GPIO Pad Data Output (GPDO165)	8	RW	00h
13A7h	SIUL2 GPIO Pad Data Output (GPDO164)	8	RW	00h
13A8h	SIUL2 GPIO Pad Data Output (GPDO171)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
13A9h	SIUL2 GPIO Pad Data Output (GPDO170)	8	RW	00h
13AAh	SIUL2 GPIO Pad Data Output (GPDO169)	8	RW	00h
13ABh	SIUL2 GPIO Pad Data Output (GPDO168)	8	RW	00h
13ACh	SIUL2 GPIO Pad Data Output (GPDO175)	8	RW	00h
13ADh	SIUL2 GPIO Pad Data Output (GPDO174)	8	RW	00h
13AEh	SIUL2 GPIO Pad Data Output (GPDO173)	8	RW	00h
13AFh	SIUL2 GPIO Pad Data Output (GPDO172)	8	RW	00h
13B0h	SIUL2 GPIO Pad Data Output (GPDO179)	8	RW	00h
13B1h	SIUL2 GPIO Pad Data Output (GPDO178)	8	RW	00h
13B2h	SIUL2 GPIO Pad Data Output (GPDO177)	8	RW	00h
13B3h	SIUL2 GPIO Pad Data Output (GPDO176)	8	RW	00h
13B4h	SIUL2 GPIO Pad Data Output (GPDO183)	8	RW	00h
13B5h	SIUL2 GPIO Pad Data Output (GPDO182)	8	RW	00h
13B6h	SIUL2 GPIO Pad Data Output (GPDO181)	8	RW	00h
13B7h	SIUL2 GPIO Pad Data Output (GPDO180)	8	RW	00h
13B8h	SIUL2 GPIO Pad Data Output (GPDO187)	8	RW	00h
13B9h	SIUL2 GPIO Pad Data Output (GPDO186)	8	RW	00h
13BAh	SIUL2 GPIO Pad Data Output (GPDO185)	8	RW	00h
13BBh	SIUL2 GPIO Pad Data Output (GPDO184)	8	RW	00h
13BCh	SIUL2 GPIO Pad Data Output (GPDO191)	8	RW	00h
13BDh	SIUL2 GPIO Pad Data Output (GPDO190)	8	RW	00h
13BEh	SIUL2 GPIO Pad Data Output (GPDO189)	8	RW	00h
13BFh	SIUL2 GPIO Pad Data Output (GPDO188)	8	RW	00h
13C0h	SIUL2 GPIO Pad Data Output (GPDO195)	8	RW	00h
13C1h	SIUL2 GPIO Pad Data Output (GPDO194)	8	RW	00h
13C2h	SIUL2 GPIO Pad Data Output (GPDO193)	8	RW	00h
13C3h	SIUL2 GPIO Pad Data Output (GPDO192)	8	RW	00h
13C4h	SIUL2 GPIO Pad Data Output (GPDO199)	8	RW	00h
13C5h	SIUL2 GPIO Pad Data Output (GPDO198)	8	RW	00h
13C6h	SIUL2 GPIO Pad Data Output (GPDO197)	8	RW	00h
13C7h	SIUL2 GPIO Pad Data Output (GPDO196)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
13C8h	SIUL2 GPIO Pad Data Output (GPDO203)	8	RW	00h
13C9h	SIUL2 GPIO Pad Data Output (GPDO202)	8	RW	00h
13CAh	SIUL2 GPIO Pad Data Output (GPDO201)	8	RW	00h
13CBh	SIUL2 GPIO Pad Data Output (GPDO200)	8	RW	00h
13CCh	SIUL2 GPIO Pad Data Output (GPDO207)	8	RW	00h
13CDh	SIUL2 GPIO Pad Data Output (GPDO206)	8	RW	00h
13CEh	SIUL2 GPIO Pad Data Output (GPDO205)	8	RW	00h
13CFh	SIUL2 GPIO Pad Data Output (GPDO204)	8	RW	00h
13D0h	SIUL2 GPIO Pad Data Output (GPDO211)	8	RW	00h
13D1h	SIUL2 GPIO Pad Data Output (GPDO210)	8	RW	00h
13D2h	SIUL2 GPIO Pad Data Output (GPDO209)	8	RW	00h
13D3h	SIUL2 GPIO Pad Data Output (GPDO208)	8	RW	00h
13D4h	SIUL2 GPIO Pad Data Output (GPDO215)	8	RW	00h
13D5h	SIUL2 GPIO Pad Data Output (GPDO214)	8	RW	00h
13D6h	SIUL2 GPIO Pad Data Output (GPDO213)	8	RW	00h
13D7h	SIUL2 GPIO Pad Data Output (GPDO212)	8	RW	00h
13D8h	SIUL2 GPIO Pad Data Output (GPDO219)	8	RW	00h
13D9h	SIUL2 GPIO Pad Data Output (GPDO218)	8	RW	00h
13DAh	SIUL2 GPIO Pad Data Output (GPDO217)	8	RW	00h
13DBh	SIUL2 GPIO Pad Data Output (GPDO216)	8	RW	00h
1500h	SIUL2 GPIO Pad Data Input (GPDI3)	8	R	00h
1501h	SIUL2 GPIO Pad Data Input (GPDI2)	8	R	00h
1502h	SIUL2 GPIO Pad Data Input (GPDI1)	8	R	00h
1503h	SIUL2 GPIO Pad Data Input (GPDI0)	8	R	00h
1504h	SIUL2 GPIO Pad Data Input (GPDI7)	8	R	00h
1505h	SIUL2 GPIO Pad Data Input (GPDI6)	8	R	00h
1506h	SIUL2 GPIO Pad Data Input (GPDI5)	8	R	00h
1507h	SIUL2 GPIO Pad Data Input (GPDI4)	8	R	00h
1508h	SIUL2 GPIO Pad Data Input (GPDI11)	8	R	00h
1509h	SIUL2 GPIO Pad Data Input (GPDI10)	8	R	00h
150Ah	SIUL2 GPIO Pad Data Input (GPDI9)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
150Bh	SIUL2 GPIO Pad Data Input (GPDI8)	8	R	00h
150Ch	SIUL2 GPIO Pad Data Input (GPDI15)	8	R	00h
150Dh	SIUL2 GPIO Pad Data Input (GPDI14)	8	R	00h
150Eh	SIUL2 GPIO Pad Data Input (GPDI13)	8	R	00h
150Fh	SIUL2 GPIO Pad Data Input (GPDI12)	8	R	00h
1510h	SIUL2 GPIO Pad Data Input (GPDI19)	8	R	00h
1511h	SIUL2 GPIO Pad Data Input (GPDI18)	8	R	00h
1512h	SIUL2 GPIO Pad Data Input (GPDI17)	8	R	00h
1513h	SIUL2 GPIO Pad Data Input (GPDI16)	8	R	00h
1514h	SIUL2 GPIO Pad Data Input (GPDI23)	8	R	00h
1515h	SIUL2 GPIO Pad Data Input (GPDI22)	8	R	00h
1516h	SIUL2 GPIO Pad Data Input (GPDI21)	8	R	00h
1517h	SIUL2 GPIO Pad Data Input (GPDI20)	8	R	00h
1518h	SIUL2 GPIO Pad Data Input (GPDI27)	8	R	00h
1519h	SIUL2 GPIO Pad Data Input (GPDI26)	8	R	00h
151Ah	SIUL2 GPIO Pad Data Input (GPDI25)	8	R	00h
151Bh	SIUL2 GPIO Pad Data Input (GPDI24)	8	R	00h
151Ch	SIUL2 GPIO Pad Data Input (GPDI31)	8	R	00h
151Dh	SIUL2 GPIO Pad Data Input (GPDI30)	8	R	00h
151Eh	SIUL2 GPIO Pad Data Input (GPDI29)	8	R	00h
151Fh	SIUL2 GPIO Pad Data Input (GPDI28)	8	R	00h
1520h	SIUL2 GPIO Pad Data Input (GPDI35)	8	R	00h
1521h	SIUL2 GPIO Pad Data Input (GPDI34)	8	R	00h
1522h	SIUL2 GPIO Pad Data Input (GPDI33)	8	R	00h
1523h	SIUL2 GPIO Pad Data Input (GPDI32)	8	R	00h
1526h	SIUL2 GPIO Pad Data Input (GPDI37)	8	R	00h
1527h	SIUL2 GPIO Pad Data Input (GPDI36)	8	R	00h
1528h	SIUL2 GPIO Pad Data Input (GPDI43)	8	R	00h
1529h	SIUL2 GPIO Pad Data Input (GPDI42)	8	R	00h
152Ah	SIUL2 GPIO Pad Data Input (GPDI41)	8	R	00h
152Bh	SIUL2 GPIO Pad Data Input (GPDI40)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
152Ch	SIUL2 GPIO Pad Data Input (GPDI47)	8	R	00h
152Dh	SIUL2 GPIO Pad Data Input (GPDI46)	8	R	00h
152Eh	SIUL2 GPIO Pad Data Input (GPDI45)	8	R	00h
152Fh	SIUL2 GPIO Pad Data Input (GPDI44)	8	R	00h
1530h	SIUL2 GPIO Pad Data Input (GPDI51)	8	R	00h
1531h	SIUL2 GPIO Pad Data Input (GPDI50)	8	R	00h
1532h	SIUL2 GPIO Pad Data Input (GPDI49)	8	R	00h
1533h	SIUL2 GPIO Pad Data Input (GPDI48)	8	R	00h
1534h	SIUL2 GPIO Pad Data Input (GPDI55)	8	R	00h
1535h	SIUL2 GPIO Pad Data Input (GPDI54)	8	R	00h
1536h	SIUL2 GPIO Pad Data Input (GPDI53)	8	R	00h
1537h	SIUL2 GPIO Pad Data Input (GPDI52)	8	R	00h
1538h	SIUL2 GPIO Pad Data Input (GPDI59)	8	R	00h
1539h	SIUL2 GPIO Pad Data Input (GPDI58)	8	R	00h
153Ah	SIUL2 GPIO Pad Data Input (GPDI57)	8	R	00h
153Bh	SIUL2 GPIO Pad Data Input (GPDI56)	8	R	00h
153Ch	SIUL2 GPIO Pad Data Input (GPDI63)	8	R	00h
153Dh	SIUL2 GPIO Pad Data Input (GPDI62)	8	R	00h
153Eh	SIUL2 GPIO Pad Data Input (GPDI61)	8	R	00h
153Fh	SIUL2 GPIO Pad Data Input (GPDI60)	8	R	00h
1540h	SIUL2 GPIO Pad Data Input (GPDI67)	8	R	00h
1541h	SIUL2 GPIO Pad Data Input (GPDI66)	8	R	00h
1542h	SIUL2 GPIO Pad Data Input (GPDI65)	8	R	00h
1543h	SIUL2 GPIO Pad Data Input (GPDI64)	8	R	00h
1544h	SIUL2 GPIO Pad Data Input (GPDI71)	8	R	00h
1545h	SIUL2 GPIO Pad Data Input (GPDI70)	8	R	00h
1546h	SIUL2 GPIO Pad Data Input (GPDI69)	8	R	00h
1547h	SIUL2 GPIO Pad Data Input (GPDI68)	8	R	00h
1548h	SIUL2 GPIO Pad Data Input (GPDI75)	8	R	00h
1549h	SIUL2 GPIO Pad Data Input (GPDI74)	8	R	00h
154Ah	SIUL2 GPIO Pad Data Input (GPDI73)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
154Bh	SIUL2 GPIO Pad Data Input (GPDI72)	8	R	00h
154Ch	SIUL2 GPIO Pad Data Input (GPDI79)	8	R	00h
154Dh	SIUL2 GPIO Pad Data Input (GPDI78)	8	R	00h
154Eh	SIUL2 GPIO Pad Data Input (GPDI77)	8	R	00h
154Fh	SIUL2 GPIO Pad Data Input (GPDI76)	8	R	00h
1550h	SIUL2 GPIO Pad Data Input (GPDI83)	8	R	00h
1551h	SIUL2 GPIO Pad Data Input (GPDI82)	8	R	00h
1552h	SIUL2 GPIO Pad Data Input (GPDI81)	8	R	00h
1553h	SIUL2 GPIO Pad Data Input (GPDI80)	8	R	00h
1554h	SIUL2 GPIO Pad Data Input (GPDI87)	8	R	00h
1555h	SIUL2 GPIO Pad Data Input (GPDI86)	8	R	00h
1556h	SIUL2 GPIO Pad Data Input (GPDI85)	8	R	00h
1557h	SIUL2 GPIO Pad Data Input (GPDI84)	8	R	00h
1558h	SIUL2 GPIO Pad Data Input (GPDI91)	8	R	00h
1559h	SIUL2 GPIO Pad Data Input (GPDI90)	8	R	00h
155Ah	SIUL2 GPIO Pad Data Input (GPDI89)	8	R	00h
155Bh	SIUL2 GPIO Pad Data Input (GPDI88)	8	R	00h
155Ch	SIUL2 GPIO Pad Data Input (GPDI95)	8	R	00h
155Dh	SIUL2 GPIO Pad Data Input (GPDI94)	8	R	00h
155Eh	SIUL2 GPIO Pad Data Input (GPDI93)	8	R	00h
155Fh	SIUL2 GPIO Pad Data Input (GPDI92)	8	R	00h
1560h	SIUL2 GPIO Pad Data Input (GPDI99)	8	R	00h
1561h	SIUL2 GPIO Pad Data Input (GPDI98)	8	R	00h
1562h	SIUL2 GPIO Pad Data Input (GPDI97)	8	R	00h
1563h	SIUL2 GPIO Pad Data Input (GPDI96)	8	R	00h
1564h	SIUL2 GPIO Pad Data Input (GPDI103)	8	R	00h
1565h	SIUL2 GPIO Pad Data Input (GPDI102)	8	R	00h
1566h	SIUL2 GPIO Pad Data Input (GPDI101)	8	R	00h
1567h	SIUL2 GPIO Pad Data Input (GPDI100)	8	R	00h
1568h	SIUL2 GPIO Pad Data Input (GPDI107)	8	R	00h
1569h	SIUL2 GPIO Pad Data Input (GPDI106)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
156Ah	SIUL2 GPIO Pad Data Input (GPDI105)	8	R	00h
156Bh	SIUL2 GPIO Pad Data Input (GPDI104)	8	R	00h
156Ch	SIUL2 GPIO Pad Data Input (GPDI111)	8	R	00h
156Dh	SIUL2 GPIO Pad Data Input (GPDI110)	8	R	00h
156Eh	SIUL2 GPIO Pad Data Input (GPDI109)	8	R	00h
156Fh	SIUL2 GPIO Pad Data Input (GPDI108)	8	R	00h
1570h	SIUL2 GPIO Pad Data Input (GPDI115)	8	R	00h
1571h	SIUL2 GPIO Pad Data Input (GPDI114)	8	R	00h
1572h	SIUL2 GPIO Pad Data Input (GPDI113)	8	R	00h
1573h	SIUL2 GPIO Pad Data Input (GPDI112)	8	R	00h
1574h	SIUL2 GPIO Pad Data Input (GPDI119)	8	R	00h
1575h	SIUL2 GPIO Pad Data Input (GPDI118)	8	R	00h
1576h	SIUL2 GPIO Pad Data Input (GPDI117)	8	R	00h
1577h	SIUL2 GPIO Pad Data Input (GPDI116)	8	R	00h
1578h	SIUL2 GPIO Pad Data Input (GPDI123)	8	R	00h
1579h	SIUL2 GPIO Pad Data Input (GPDI122)	8	R	00h
157Ah	SIUL2 GPIO Pad Data Input (GPDI121)	8	R	00h
157Bh	SIUL2 GPIO Pad Data Input (GPDI120)	8	R	00h
157Ch	SIUL2 GPIO Pad Data Input (GPDI127)	8	R	00h
157Dh	SIUL2 GPIO Pad Data Input (GPDI126)	8	R	00h
157Eh	SIUL2 GPIO Pad Data Input (GPDI125)	8	R	00h
157Fh	SIUL2 GPIO Pad Data Input (GPDI124)	8	R	00h
1580h	SIUL2 GPIO Pad Data Input (GPDI131)	8	R	00h
1581h	SIUL2 GPIO Pad Data Input (GPDI130)	8	R	00h
1582h	SIUL2 GPIO Pad Data Input (GPDI129)	8	R	00h
1583h	SIUL2 GPIO Pad Data Input (GPDI128)	8	R	00h
1584h	SIUL2 GPIO Pad Data Input (GPDI135)	8	R	00h
1585h	SIUL2 GPIO Pad Data Input (GPDI134)	8	R	00h
1586h	SIUL2 GPIO Pad Data Input (GPDI133)	8	R	00h
1587h	SIUL2 GPIO Pad Data Input (GPDI132)	8	R	00h
1588h	SIUL2 GPIO Pad Data Input (GPDI139)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1589h	SIUL2 GPIO Pad Data Input (GPDI138)	8	R	00h
158Ah	SIUL2 GPIO Pad Data Input (GPDI137)	8	R	00h
158Bh	SIUL2 GPIO Pad Data Input (GPDI136)	8	R	00h
158Ch	SIUL2 GPIO Pad Data Input (GPDI143)	8	R	00h
158Dh	SIUL2 GPIO Pad Data Input (GPDI142)	8	R	00h
158Eh	SIUL2 GPIO Pad Data Input (GPDI141)	8	R	00h
158Fh	SIUL2 GPIO Pad Data Input (GPDI140)	8	R	00h
1590h	SIUL2 GPIO Pad Data Input (GPDI147)	8	R	00h
1591h	SIUL2 GPIO Pad Data Input (GPDI146)	8	R	00h
1592h	SIUL2 GPIO Pad Data Input (GPDI145)	8	R	00h
1593h	SIUL2 GPIO Pad Data Input (GPDI144)	8	R	00h
1594h	SIUL2 GPIO Pad Data Input (GPDI151)	8	R	00h
1595h	SIUL2 GPIO Pad Data Input (GPDI150)	8	R	00h
1596h	SIUL2 GPIO Pad Data Input (GPDI149)	8	R	00h
1597h	SIUL2 GPIO Pad Data Input (GPDI148)	8	R	00h
1598h	SIUL2 GPIO Pad Data Input (GPDI155)	8	R	00h
1599h	SIUL2 GPIO Pad Data Input (GPDI154)	8	R	00h
159Ah	SIUL2 GPIO Pad Data Input (GPDI153)	8	R	00h
159Bh	SIUL2 GPIO Pad Data Input (GPDI152)	8	R	00h
159Ch	SIUL2 GPIO Pad Data Input (GPDI159)	8	R	00h
159Dh	SIUL2 GPIO Pad Data Input (GPDI158)	8	R	00h
159Eh	SIUL2 GPIO Pad Data Input (GPDI157)	8	R	00h
159Fh	SIUL2 GPIO Pad Data Input (GPDI156)	8	R	00h
15A0h	SIUL2 GPIO Pad Data Input (GPDI163)	8	R	00h
15A1h	SIUL2 GPIO Pad Data Input (GPDI162)	8	R	00h
15A2h	SIUL2 GPIO Pad Data Input (GPDI161)	8	R	00h
15A3h	SIUL2 GPIO Pad Data Input (GPDI160)	8	R	00h
15A4h	SIUL2 GPIO Pad Data Input (GPDI167)	8	R	00h
15A5h	SIUL2 GPIO Pad Data Input (GPDI166)	8	R	00h
15A6h	SIUL2 GPIO Pad Data Input (GPDI165)	8	R	00h
15A7h	SIUL2 GPIO Pad Data Input (GPDI164)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
15A8h	SIUL2 GPIO Pad Data Input (GPDI171)	8	R	00h
15A9h	SIUL2 GPIO Pad Data Input (GPDI170)	8	R	00h
15AAh	SIUL2 GPIO Pad Data Input (GPDI169)	8	R	00h
15ABh	SIUL2 GPIO Pad Data Input (GPDI168)	8	R	00h
15ACh	SIUL2 GPIO Pad Data Input (GPDI175)	8	R	00h
15ADh	SIUL2 GPIO Pad Data Input (GPDI174)	8	R	00h
15AEh	SIUL2 GPIO Pad Data Input (GPDI173)	8	R	00h
15AFh	SIUL2 GPIO Pad Data Input (GPDI172)	8	R	00h
15B0h	SIUL2 GPIO Pad Data Input (GPDI179)	8	R	00h
15B1h	SIUL2 GPIO Pad Data Input (GPDI178)	8	R	00h
15B2h	SIUL2 GPIO Pad Data Input (GPDI177)	8	R	00h
15B3h	SIUL2 GPIO Pad Data Input (GPDI176)	8	R	00h
15B4h	SIUL2 GPIO Pad Data Input (GPDI183)	8	R	00h
15B5h	SIUL2 GPIO Pad Data Input (GPDI182)	8	R	00h
15B6h	SIUL2 GPIO Pad Data Input (GPDI181)	8	R	00h
15B7h	SIUL2 GPIO Pad Data Input (GPDI180)	8	R	00h
15B8h	SIUL2 GPIO Pad Data Input (GPDI187)	8	R	00h
15B9h	SIUL2 GPIO Pad Data Input (GPDI186)	8	R	00h
15BAh	SIUL2 GPIO Pad Data Input (GPDI185)	8	R	00h
15BBh	SIUL2 GPIO Pad Data Input (GPDI184)	8	R	00h
15BCh	SIUL2 GPIO Pad Data Input (GPDI191)	8	R	00h
15BDh	SIUL2 GPIO Pad Data Input (GPDI190)	8	R	00h
15BEh	SIUL2 GPIO Pad Data Input (GPDI189)	8	R	00h
15BFh	SIUL2 GPIO Pad Data Input (GPDI188)	8	R	00h
15C0h	SIUL2 GPIO Pad Data Input (GPDI195)	8	R	00h
15C1h	SIUL2 GPIO Pad Data Input (GPDI194)	8	R	00h
15C2h	SIUL2 GPIO Pad Data Input (GPDI193)	8	R	00h
15C3h	SIUL2 GPIO Pad Data Input (GPDI192)	8	R	00h
15C4h	SIUL2 GPIO Pad Data Input (GPDI199)	8	R	00h
15C5h	SIUL2 GPIO Pad Data Input (GPDI198)	8	R	00h
15C6h	SIUL2 GPIO Pad Data Input (GPDI197)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
15C7h	SIUL2 GPIO Pad Data Input (GPDI196)	8	R	00h
15C8h	SIUL2 GPIO Pad Data Input (GPDI203)	8	R	00h
15C9h	SIUL2 GPIO Pad Data Input (GPDI202)	8	R	00h
15CAh	SIUL2 GPIO Pad Data Input (GPDI201)	8	R	00h
15CBh	SIUL2 GPIO Pad Data Input (GPDI200)	8	R	00h
15CCh	SIUL2 GPIO Pad Data Input (GPDI207)	8	R	00h
15CDh	SIUL2 GPIO Pad Data Input (GPDI206)	8	R	00h
15CEh	SIUL2 GPIO Pad Data Input (GPDI205)	8	R	00h
15CFh	SIUL2 GPIO Pad Data Input (GPDI204)	8	R	00h
15D0h	SIUL2 GPIO Pad Data Input (GPDI211)	8	R	00h
15D1h	SIUL2 GPIO Pad Data Input (GPDI210)	8	R	00h
15D2h	SIUL2 GPIO Pad Data Input (GPDI209)	8	R	00h
15D3h	SIUL2 GPIO Pad Data Input (GPDI208)	8	R	00h
15D4h	SIUL2 GPIO Pad Data Input (GPDI215)	8	R	00h
15D5h	SIUL2 GPIO Pad Data Input (GPDI214)	8	R	00h
15D6h	SIUL2 GPIO Pad Data Input (GPDI213)	8	R	00h
15D7h	SIUL2 GPIO Pad Data Input (GPDI212)	8	R	00h
15D8h	SIUL2 GPIO Pad Data Input (GPDI219)	8	R	00h
15D9h	SIUL2 GPIO Pad Data Input (GPDI218)	8	R	00h
15DAh	SIUL2 GPIO Pad Data Input (GPDI217)	8	R	00h
15DBh	SIUL2 GPIO Pad Data Input (GPDI216)	8	R	00h
1700h - 1704h	SIUL2 Parallel GPIO Pad Data Out (PGPDO0 - PGPDO3) [1]	16	RW	0000h
1706h	SIUL2 Parallel GPIO Pad Data Out (PGPDO2)	16	RW	0000h
1708h - 1716h	SIUL2 Parallel GPIO Pad Data Out (PGPDO4 - PGPDO11) [1]	16	RW	0000h
1718h	SIUL2 Parallel GPIO Pad Data Out (PGPDO13)	16	RW	0000h
171Ah	SIUL2 Parallel GPIO Pad Data Out (PGPDO12)	16	RW	0000h
1740h - 1744h	SIUL2 Parallel GPIO Pad Data In (PGPDI0 - PGPDI3) [1]	16	R	0000h
1746h	SIUL2 Parallel GPIO Pad Data In (PGPDI2)	16	R	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1748h - 1756h	SIUL2 Parallel GPIO Pad Data In (PGPDI4 - PGPDI11) [1]	16	R	0000h
1758h	SIUL2 Parallel GPIO Pad Data In (PGPDI13)	16	R	0000h
175Ah	SIUL2 Parallel GPIO Pad Data In (PGPDI12)	16	R	0000h
1780h - 1784h	SIUL2 Masked Parallel GPIO Pad Data Out (MPGPDO0 - MPGPDO1)	32	W	0000_0000h
1788h	SIUL2 Masked Parallel GPIO Pad Data Out (MPGPDO2)	32	W	0000_0000h
178Ch - 17B0h	SIUL2 Masked Parallel GPIO Pad Data Out (MPGPDO3 - MPGPDO12)	32	W	0000_0000h
17B4h	SIUL2 Masked Parallel GPIO Pad Data Out (MPGPDO13)	32	W	0000_0000h

1. In this array, the index and offset values of the registers do not increment in direct alignment. For details, see the register description.

9.6.2 SIUL2 MCU ID Register #1 (MIDR1)

Offset

Register	Offset
MIDR1	4h

Function

This register holds identification information about the device.

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram

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

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

Fields

Field	Function
31-26 PRODUCT_LINE_LETTER	Product Line Letter Identified the ASCII character in MCU Part Number. This field specifies the part number suffix, and needs to be combined with MIDR1[PART_NO] to provide the full chip number. 0x0B K This value is set at the factory and cannot be changed. All other values are reserved.
25-16 PART_NO	MCU Part Number MWCT2014S - 0x136 MWCT2015S - 0x137 MWCT2016S - 0x138 MWCT2D16S - 0x142 MWCT2D17S - 0x144
15-8 —	Reserved
7-4 MAJOR_MASK	Major Mask Revision
3-0 MINOR_MASK	Minor Mask Revision

9.6.3 SIUL2 MCU ID Register #2 (MIDR2)

Offset

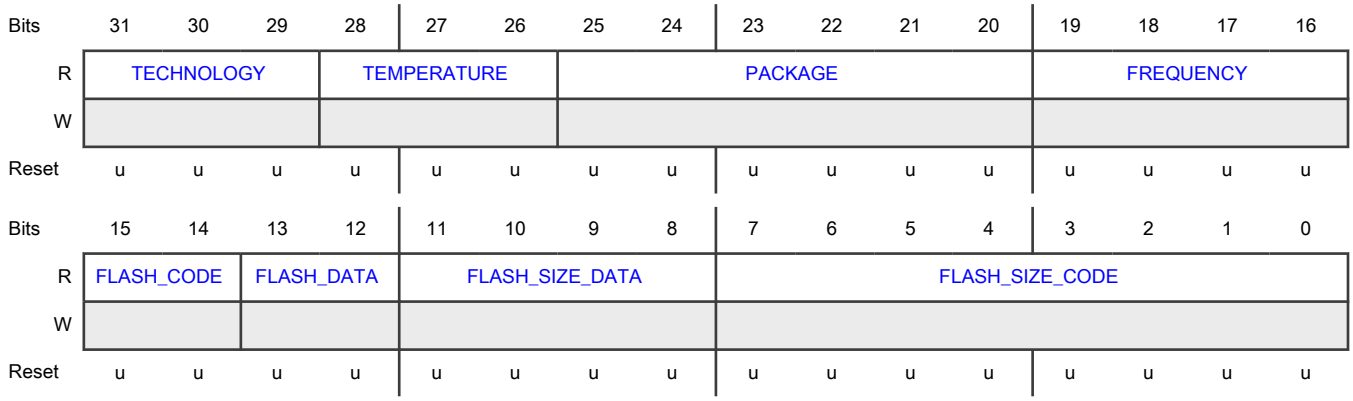
Register	Offset
MIDR2	8h

Function

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram



Fields

Field	Function
31-29 TECHNOLOGY	Technology Identifies the silicon technology. 001b - C40EFS3
28-26 TEMPERATURE	Temperature Identifies the ambient temperature range. 100b - M = 125C
25-20 PACKAGE	Package This field can by read by software to determine the package type that is used for the particular device. 10_0010b - 100-HDQFP 10_0011b - 100-HDQFP 10_0101b - 172-HDQFP 10_0110b - 172-HDQFP 10_0111b - 172-HDQFP-EP
19-16 FREQUENCY	Frequency Identifies maximum core frequency. Qualified by Product Line Letter to provide wider range of frequencies. 0011b - 120 MHz 0100b - 160 MHz
15-14 FLASH_CODE	Flash Code Identifies the location of Code Flash, if any, within the package. 10b - Monolithic
13-12	Flash Data

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLASH_DATA	Identifies the location of Data Flash, if any, within the package. 10b - Monolithic
11-8 FLASH_SIZE_DATA	Flash Size Data Identifies the Flash (EE) memory size. 0000b - 64KB 0001b - 128KB 0010b - 256KB
7-0 FLASH_SIZE_CODE	Flash Size Code Identifies the Flash (code) memory size. 0000_0010b - 512kB 0000_0100b - 1MB 0000_1000b - 2.00MB 0000_1100b - 3.00MB 0001_0000b - 4.00MB 0001_1000b - 6.00MB 0010_0000b - 8.00MB

9.6.4 SIUL2 DMA/Interrupt Status Flag 0 (DISR0)

Offset

Register	Offset
DISR0	10h

Function

Contains flag bits that record an event on the external IRQ pins. When an event as defined in IREER0 and IFEER0 occurs, the corresponding flag bit is set. The IRQ flag bit is set regardless of the state of the corresponding DIRER0[EIRE*n*] field. The IRQ flag bit remains set until you write 0 to it or the servicing of a DMA request writes a 0 to the field. The IRQ flag bits are cleared by writing a 1 to the bits. A write of 0 has no effect.

This register supports 8-, 16-, and 32-bit accesses.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EIF31	EIF30	EIF29	EIF28	EIF27	EIF26	EIF25	EIF24	EIF23	EIF22	EIF21	EIF20	EIF19	EIF18	EIF17	EIF16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EIF15	EIF14	EIF13	EIF12	EIF11	EIF10	EIF9	EIF8	EIF7	EIF6	EIF5	EIF4	EIF3	EIF2	EIF1	EIF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EIF31	External Interrupt Status Flag 31 If enabled (DIRERR31 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER31 and IFEER31 has occurred.
30 EIF30	External Interrupt Status Flag 30 If enabled (DIRERR30 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER30 and IFEER30 has occurred.
29 EIF29	External Interrupt Status Flag 29 If enabled (DIRERR29 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER29 and IFEER29 has occurred.
28 EIF28	External Interrupt Status Flag 28 If enabled (DIRERR28 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER28 and IFEER28 has occurred.
27 EIF27	External Interrupt Status Flag 27 If enabled (DIRERR27 = 1) causes an interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER27 and IFEER27 has occurred.
26 EIF26	External Interrupt Status Flag 26 If enabled (DIRERR26 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER26 and IFEER26 has occurred.
25 EIF25	External Interrupt Status Flag 25 If enabled (DIRERR25 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER25 and IFEER25 has occurred.
24 EIF24	External Interrupt Status Flag 24 If enabled (DIRERR24 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER24 and IFEER24 has occurred.
23 EIF23	External Interrupt Status Flag 23 If enabled (DIRERR23 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER23 and IFEER23 has occurred.
22 EIF22	External Interrupt Status Flag 22 If enabled (DIRERR22 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER22 and IFEER22 has occurred.
21 EIF21	External Interrupt Status Flag 21 If enabled (DIRERR21 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER21 and IFEER21 has occurred.
20 EIF20	External Interrupt Status Flag 20 If enabled (DIRERR20 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER20 and IFEER20 has occurred.
19 EIF19	External Interrupt Status Flag 19 If enabled (DIRERR19 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER19 and IFEER19 has occurred.
18 EIF18	External Interrupt Status Flag 18 If enabled (DIRERR18 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER18 and IFEER18 has occurred.
17 EIF17	External Interrupt Status Flag 17 If enabled (DIRERR17 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER17 and IFEER17 has occurred.
16 EIF16	External Interrupt Status Flag 16 If enabled (DIRERR16 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER16 and IFEER16 has occurred.
15 EIF15	External Interrupt Status Flag 15 If enabled (DIRERR15 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - An interrupt event as defined by IREER15 and IFEER15 has occurred.
14 EIF14	External Interrupt Status Flag 14 If enabled (DIRERR14 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER14 and IFEER14 has occurred.
13 EIF13	External Interrupt Status Flag 13 If enabled (DIRERR13 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER13 and IFEER13 has occurred.
12 EIF12	External Interrupt Status Flag 12 If enabled (DIRERR12 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER12 and IFEER12 has occurred.
11 EIF11	External Interrupt Status Flag 11 If enabled (DIRERR11 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER11 and IFEER11 has occurred.
10 EIF10	External Interrupt Status Flag 10 If enabled (DIRERR10 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER10 and IFEER10 has occurred.
9 EIF9	External Interrupt Status Flag 9 If enabled (DIRERR9 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER9 and IFEER9 has occurred.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 EIF8	External Interrupt Status Flag 8 If enabled (DIRERR8 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER8 and IFEER8 has occurred.
7 EIF7	External Interrupt Status Flag 7 If enabled (DIRERR7 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER7 and IFEER7 has occurred.
6 EIF6	External Interrupt Status Flag 6 If enabled (DIRERR6 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER6 and IFEER6 has occurred.
5 EIF5	External Interrupt Status Flag 5 If enabled (DIRERR5 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER5 and IFEER5 has occurred.
4 EIF4	External Interrupt Status Flag 4 If enabled (DIRERR4 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER4 and IFEER4 has occurred.
3 EIF3	External Interrupt Status Flag 3 If enabled (DIRERR3 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER3 and IFEER3 has occurred.
2	External Interrupt Status Flag 2 If enabled (DIRERR2 = 1) causes an interrupt or DMA request.

Table continues on the next page...

Table continued from the previous page...

Field	Function
EIF2	This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER2 and IFEER2 has occurred.
1 EIF1	External Interrupt Status Flag 1 If enabled (DIRERR1 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER1 and IFEER1 has occurred.
0 EIF0	External Interrupt Status Flag 0 If enabled (DIRERR0 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER0 and IFEER0 has occurred.

9.6.5 SIUL2 DMA/Interrupt Request Enable 0 (DIRER0)

Offset

Register	Offset
DIRER0	18h

Function

Enables the assertion of DMA or interrupt request to the interrupt controller if the corresponding DISR0[EIF*n*] bit is set. The type of request is determined by the corresponding DIRSR0[DIRSR*n*] bit.

This register supports 8-, 16-, and 32-bit accesses.

NOTE

Once DIRSR0 selects a DMA request, you cannot enable or disable it.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EIRE31	EIRE30	EIRE29	EIRE28	EIRE27	EIRE26	EIRE25	EIRE24	EIRE23	EIRE22	EIRE21	EIRE20	EIRE19	EIRE18	EIRE17	EIRE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EIRE15	EIRE14	EIRE13	EIRE12	EIRE11	EIRE10	EIRE9	EIRE8	EIRE7	EIRE6	EIRE5	EIRE4	EIRE3	EIRE2	EIRE1	EIRE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EIRE31	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
30 EIRE30	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
29 EIRE29	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
28 EIRE28	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
27 EIRE27	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
26 EIRE26	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
25 EIRE25	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
24 EIRE24	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
23 EIRE23	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
22 EIRE22	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
21 EIRE21	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
20 EIRE20	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
19 EIRE19	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
18	External Interrupt Request Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
EIRE18	Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
17 EIRE17	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
16 EIRE16	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
15 EIRE15	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
14 EIRE14	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
13 EIRE13	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
12 EIRE12	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
11 EIRE11	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 EIRE10	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
9 EIRE9	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
8 EIRE8	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
7 EIRE7	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
6 EIRE6	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
5 EIRE5	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
4 EIRE4	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
3 EIRE3	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
2 EIRE2	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
1 EIRE1	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
0 EIRE0	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled

9.6.6 SIUL2 DMA/Interrupt Request Select 0 (DIRSR0)

Offset

Register	Offset
DIRSR0	20h

Function

Selects between the DMA or interrupt request. The DIRSR0[DIRSR*n*] bit determines whether DMA or an interrupt request asserts the corresponding DISR0[EIF*n*] bit.

This register supports 8-, 16-, and 32-bit accesses.

Diagram

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

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

Fields

Field	Function
31 DIRSR31	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
30 DIRSR30	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
29 DIRSR29	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
28 DIRSR28	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
27 DIRSR27	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
26 DIRSR26	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
25 DIRSR25	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt request 1b - Reserved
24 DIRSR24	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
23 DIRSR23	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
22 DIRSR22	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
21 DIRSR21	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
20 DIRSR20	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
19 DIRSR19	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
18	DMA/Interrupt Request Select Register

Table continues on the next page...

Table continued from the previous page...

Field	Function
DIRSR18	Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
17 DIRSR17	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
16 DIRSR16	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
15 DIRSR15	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
14 DIRSR14	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
13 DIRSR13	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
12 DIRSR12	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 DIRSR11	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
10 DIRSR10	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
9 DIRSR9	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
8 DIRSR8	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
7 DIRSR7	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
6 DIRSR6	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
5 DIRSR5	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt request 1b - DMA request
4 DIRSR4	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
3 DIRSR3	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
2 DIRSR2	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
1 DIRSR1	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
0 DIRSR0	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request

9.6.7 SIUL2 Interrupt Rising-Edge Event Enable 0 (IREER0)

Offset

Register	Offset
IREER0	28h

Function

Enables the rising-edge triggered events on the corresponding interrupt pads.

This register supports 8-, 16-, and 32-bit accesses.

NOTE

If both the IREE and IFEE bits are cleared for the same interrupt source, the interrupt status flag for the corresponding external interrupt will never be set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IREE31	IREE30	IREE29	IREE28	IREE27	IREE26	IREE25	IREE24	IREE23	IREE22	IREE21	IREE20	IREE19	IREE18	IREE17	IREE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IREE15	IREE14	IREE13	IREE12	IREE11	IREE10	IREE9	IREE8	IREE7	IREE6	IREE5	IREE4	IREE3	IREE2	IREE1	IREE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 IREE31	Enables rising-edge events to set DISR0[EIF31]. 0b - Disabled 1b - Enabled
30 IREE30	Enables rising-edge events to set DISR0[EIF30]. 0b - Disabled 1b - Enabled
29 IREE29	Enables rising-edge events to set DISR0[EIF29]. 0b - Disabled 1b - Enabled
28 IREE28	Enables rising-edge events to set DISR0[EIF28]. 0b - Disabled 1b - Enabled
27 IREE27	Enables rising-edge events to set DISR0[EIF27]. 0b - Disabled 1b - Enabled
26	Enables rising-edge events to set DISR0[EIF26].

Table continues on the next page...

Table continued from the previous page...

Field	Function
IREE26	0b - Disabled 1b - Enabled
25 IREE25	Enables rising-edge events to set DISR0[EIF25]. 0b - Disabled 1b - Enabled
24 IREE24	Enables rising-edge events to set DISR0[EIF24]. 0b - Disabled 1b - Enabled
23 IREE23	Enables rising-edge events to set DISR0[EIF23]. 0b - Disabled 1b - Enabled
22 IREE22	Enables rising-edge events to set DISR0[EIF22]. 0b - Disabled 1b - Enabled
21 IREE21	Enables rising-edge events to set DISR0[EIF21]. 0b - Disabled 1b - Enabled
20 IREE20	Enables rising-edge events to set DISR0[EIF20]. 0b - Disabled 1b - Enabled
19 IREE19	Enables rising-edge events to set DISR0[EIF19]. 0b - Disabled 1b - Enabled
18 IREE18	Enables rising-edge events to set DISR0[EIF18]. 0b - Disabled 1b - Enabled
17 IREE17	Enables rising-edge events to set DISR0[EIF17]. 0b - Disabled 1b - Enabled
16 IREE16	Enables rising-edge events to set DISR0[EIF16]. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 IREE15	Enables rising-edge events to set DISR0[EIF15]. 0b - Disabled 1b - Enabled
14 IREE14	Enables rising-edge events to set DISR0[EIF14]. 0b - Disabled 1b - Enabled
13 IREE13	Enables rising-edge events to set DISR0[EIF13]. 0b - Disabled 1b - Enabled
12 IREE12	Enables rising-edge events to set DISR0[EIF12]. 0b - Disabled 1b - Enabled
11 IREE11	Enables rising-edge events to set DISR0[EIF11]. 0b - Disabled 1b - Enabled
10 IREE10	Enables rising-edge events to set DISR0[EIF10]. 0b - Disabled 1b - Enabled
9 IREE9	Enables rising-edge events to set DISR0[EIF9]. 0b - Disabled 1b - Enabled
8 IREE8	Enables rising-edge events to set DISR0[EIF8]. 0b - Disabled 1b - Enabled
7 IREE7	Enables rising-edge events to set DISR0[EIF7]. 0b - Disabled 1b - Enabled
6 IREE6	Enables rising-edge events to set DISR0[EIF6]. 0b - Disabled 1b - Enabled
5 IREE5	Enables rising-edge events to set DISR0[EIF5].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
4 IREE4	Enables rising-edge events to set DISR0[EIF4]. 0b - Disabled 1b - Enabled
3 IREE3	Enables rising-edge events to set DISR0[EIF3]. 0b - Disabled 1b - Enabled
2 IREE2	Enables rising-edge events to set DISR0[EIF2]. 0b - Disabled 1b - Enabled
1 IREE1	Enables rising-edge events to set DISR0[EIF1]. 0b - Disabled 1b - Enabled
0 IREE0	Enables rising-edge events to set DISR0[EIF0]. 0b - Disabled 1b - Enabled

9.6.8 SIUL2 Interrupt Falling-Edge Event Enable 0 (IFEER0)

Offset

Register	Offset
IFEER0	30h

Function

Enables falling-edge triggered events on the corresponding interrupt pads.

This register supports 8-, 16-, and 32-bit accesses.

NOTE

If both the IREE and IFEE bits are cleared for the same interrupt source, the interrupt status flag for the corresponding external interrupt will never be set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IFEE31	IFEE30	IFEE29	IFEE28	IFEE27	IFEE26	IFEE25	IFEE24	IFEE23	IFEE22	IFEE21	IFEE20	IFEE19	IFEE18	IFEE17	IFEE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IFEE15	IFEE14	IFEE13	IFEE12	IFEE11	IFEE10	IFEE9	IFEE8	IFEE7	IFEE6	IFEE5	IFEE4	IFEE3	IFEE2	IFEE1	IFEE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 IFEE31	Enables falling-edge events to set DISR0[EIF31]. 0b - Disabled 1b - Enabled
30 IFEE30	Enables falling-edge events to set DISR0[EIF30]. 0b - Disabled 1b - Enabled
29 IFEE29	Enables falling-edge events to set DISR0[EIF29]. 0b - Disabled 1b - Enabled
28 IFEE28	Enables falling-edge events to set DISR0[EIF28]. 0b - Disabled 1b - Enabled
27 IFEE27	Enables falling-edge events to set DISR0[EIF27]. 0b - Disabled 1b - Enabled
26 IFEE26	Enables falling-edge events to set DISR0[EIF26]. 0b - Disabled 1b - Enabled
25 IFEE25	Enables falling-edge events to set DISR0[EIF25]. 0b - Disabled 1b - Enabled
24	Enables falling-edge events to set DISR0[EIF24].

Table continues on the next page...

Table continued from the previous page...

Field	Function
IFEE24	0b - Disabled 1b - Enabled
23 IFEE23	Enables falling-edge events to set DISR0[EIF23]. 0b - Disabled 1b - Enabled
22 IFEE22	Enables falling-edge events to set DISR0[EIF22]. 0b - Disabled 1b - Enabled
21 IFEE21	Enables falling-edge events to set DISR0[EIF21]. 0b - Disabled 1b - Enabled
20 IFEE20	Enables falling-edge events to set DISR0[EIF20]. 0b - Disabled 1b - Enabled
19 IFEE19	Enables falling-edge events to set DISR0[EIF19]. 0b - Disabled 1b - Enabled
18 IFEE18	Enables falling-edge events to set DISR0[EIF18]. 0b - Disabled 1b - Enabled
17 IFEE17	Enables falling-edge events to set DISR0[EIF17]. 0b - Disabled 1b - Enabled
16 IFEE16	Enables falling-edge events to set DISR0[EIF16]. 0b - Disabled 1b - Enabled
15 IFEE15	Enables falling-edge events to set DISR0[EIF15]. 0b - Disabled 1b - Enabled
14 IFEE14	Enables falling-edge events to set DISR0[EIF14]. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 IFEE13	Enables falling-edge events to set DISR0[EIF13]. 0b - Disabled 1b - Enabled
12 IFEE12	Enables falling-edge events to set DISR0[EIF12]. 0b - Disabled 1b - Enabled
11 IFEE11	Enables falling-edge events to set DISR0[EIF11]. 0b - Disabled 1b - Enabled
10 IFEE10	Enables falling-edge events to set DISR0[EIF10]. 0b - Disabled 1b - Enabled
9 IFEE9	Enables falling-edge events to set DISR0[EIF9]. 0b - Disabled 1b - Enabled
8 IFEE8	Enables falling-edge events to set DISR0[EIF8]. 0b - Disabled 1b - Enabled
7 IFEE7	Enables falling-edge events to set DISR0[EIF7]. 0b - Disabled 1b - Enabled
6 IFEE6	Enables falling-edge events to set DISR0[EIF6]. 0b - Disabled 1b - Enabled
5 IFEE5	Enables falling-edge events to set DISR0[EIF5]. 0b - Disabled 1b - Enabled
4 IFEE4	Enables falling-edge events to set DISR0[EIF4]. 0b - Disabled 1b - Enabled
3 IFEE3	Enables falling-edge events to set DISR0[EIF3].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
2 IFEE2	Enables falling-edge events to set DISR0[EIF2]. 0b - Disabled 1b - Enabled
1 IFEE1	Enables falling-edge events to set DISR0[EIF1]. 0b - Disabled 1b - Enabled
0 IFEE0	Enables falling-edge events to set DISR0[EIF0]. 0b - Disabled 1b - Enabled

9.6.9 SIUL2 Interrupt Filter Enable 0 (IFER0)

Offset

Register	Offset
IFER0	38h

Function

Enables a digital filter counter on the corresponding interrupt pads to filter out glitches on the inputs.

This register supports 8-, 16-, and 32-bit accesses.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IFE31	IFE30	IFE29	IFE28	IFE27	IFE26	IFE25	IFE24	IFE23	IFE22	IFE21	IFE20	IFE19	IFE18	IFE17	IFE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IFE15	IFE14	IFE13	IFE12	IFE11	IFE10	IFE9	IFE8	IFE7	IFE6	IFE5	IFE4	IFE3	IFE2	IFE1	IFE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 IFE31	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
30 IFE30	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
29 IFE29	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
28 IFE28	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
27 IFE27	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
26 IFE26	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
25 IFE25	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
24 IFE24	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
23 IFE23	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
22 IFE22	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
21 IFE21	Enables digital glitch filter on the interrupt pad input.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
20 IFE20	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
19 IFE19	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
18 IFE18	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
17 IFE17	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
16 IFE16	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
15 IFE15	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
14 IFE14	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
13 IFE13	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
12 IFE12	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
11 IFE11	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 IFE10	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
9 IFE9	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
8 IFE8	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
7 IFE7	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
6 IFE6	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
5 IFE5	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
4 IFE4	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
3 IFE3	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
2 IFE2	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
1 IFE1	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
0	Enables digital glitch filter on the interrupt pad input.

Table continues on the next page...

Table continued from the previous page...

Field	Function
IFE0	0b - Disabled 1b - Enabled

9.6.10 SIUL2 Interrupt Filter Maximum Counter (IFMCR0 - IFMCR31)

Offset

For a = 0 to 31:

Register	Offset
IFMCRa	40h + (a × 4h)

Function

Configure the filter counter associated with each digital glitch filter.

NOTE

These registers support only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram

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

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

Fields

Field	Function
31-4 —	Reserved
3-0 MAXCNT	Maximum Interrupt Filter Counter setting MAXCNT can be 0d to 15d.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• A value of 0d, 1d, or 2d sets the filter as an all pass filter. • A value of 3d to 15d sets the filter period to TCK × MAXCNT + *n* × TCK, where: — *n* is 1, 2, 3, or 4. — TCK is the prescaled filter clock period, which is the IRC clock prescaled to the IFCP value specified in IFCPR. n accounts for the uncertainty factor in filter period calculation.

9.6.11 SIUL2 MCU ID Register #3 (MIDR3)

Offset

Register	Offset
MIDR3	200h

Function

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram

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

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

Fields

Field	Function
31-26 PROD_FAM_LET	Product Family Letter Identifies the product family letter. 01_0011b - S

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-16 PROD_FAM_NO	Product Family Number Identifies the product family number. 00_0010_0000b - 32
15-10 PART_NO_SUF	Part Number Suffix Describes the part number suffix. 00_0000b - None
9-6 —	Reserved
5-0 SYS_RAM_SIZE	System RAM Size Total RAM size in SoC, including TCMs. 00_0010b - 128kB 00_0011b - 192kB 00_0100b - 256kB 00_0110b - 512kB

9.6.12 SIUL2 MCU ID Register #4 (MIDR4)

Offset

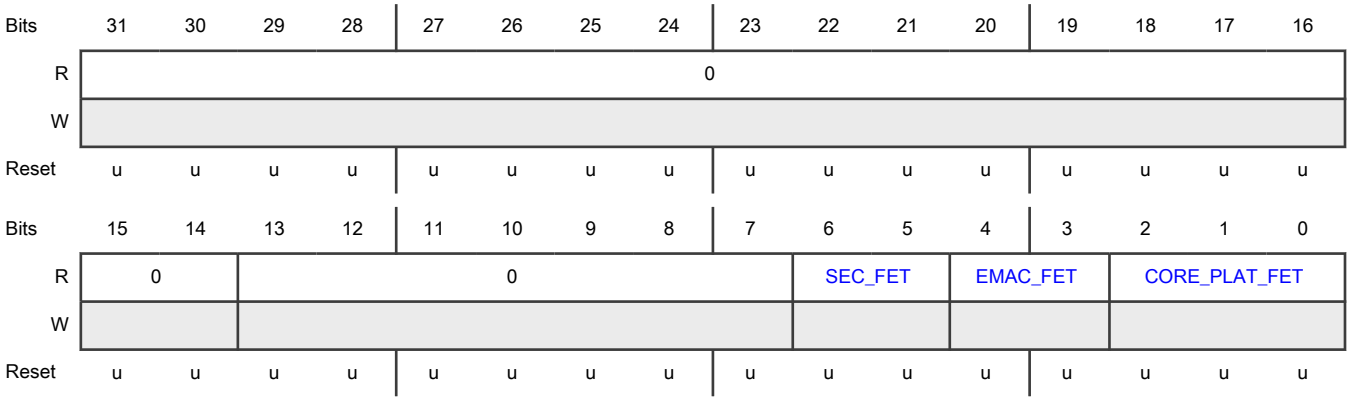
Register	Offset
MIDR4	204h

Function

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-14 —	Reserved
13-7 —	Reserved
6-5 SEC_FET	Security Feature Security feature. 01b - HSE-B
4-3 EMAC_FET	Ethernet Feature Ethernet feature. 00b - No Ethernet 01b - Ethernet present
2-0 CORE_PLAT_FET	Core Platform Options Feature Core platform options feature. 000b - Single core 001b - Dual core 010b - 1x lock-step core 011b - 1x lock-step core + 1 perf. core 100b - Triple core

9.6.13 SIUL2 Multiplexed Signal Configuration Register (MSCR0 - MSCR219)

Offset

Register	Offset
MSCR0	240h
MSCR1	244h
MSCR2	248h
MSCR3	24Ch
MSCR4	250h
MSCR5	254h
MSCR6	258h
MSCR7	25Ch
MSCR8	260h
MSCR9	264h
MSCR10	268h
MSCR11	26Ch
MSCR12	270h
MSCR13	274h
MSCR14	278h
MSCR15	27Ch
MSCR16	280h
MSCR17	284h
MSCR18	288h
MSCR19	28Ch
MSCR20	290h
MSCR21	294h
MSCR22	298h
MSCR23	29Ch
MSCR24	2A0h
MSCR25	2A4h
MSCR26	2A8h
MSCR27	2ACh
MSCR28	2B0h
MSCR29	2B4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR30	2B8h
MSCR31	2BCh
MSCR32	2C0h
MSCR33	2C4h
MSCR34	2C8h
MSCR35	2CCh
MSCR36	2D0h
MSCR37	2D4h
MSCR40	2E0h
MSCR41	2E4h
MSCR42	2E8h
MSCR43	2ECh
MSCR44	2F0h
MSCR45	2F4h
MSCR46	2F8h
MSCR47	2FCh
MSCR48	300h
MSCR49	304h
MSCR50	308h
MSCR51	30Ch
MSCR52	310h
MSCR53	314h
MSCR54	318h
MSCR55	31Ch
MSCR56	320h
MSCR57	324h
MSCR58	328h
MSCR59	32Ch
MSCR60	330h
MSCR61	334h
MSCR62	338h
MSCR63	33Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR64	340h
MSCR65	344h
MSCR66	348h
MSCR67	34Ch
MSCR68	350h
MSCR69	354h
MSCR70	358h
MSCR71	35Ch
MSCR72	360h
MSCR73	364h
MSCR74	368h
MSCR75	36Ch
MSCR76	370h
MSCR77	374h
MSCR78	378h
MSCR79	37Ch
MSCR80	380h
MSCR81	384h
MSCR82	388h
MSCR83	38Ch
MSCR84	390h
MSCR85	394h
MSCR86	398h
MSCR87	39Ch
MSCR88	3A0h
MSCR89	3A4h
MSCR90	3A8h
MSCR91	3ACh
MSCR92	3B0h
MSCR93	3B4h
MSCR94	3B8h
MSCR95	3BCh

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR96	3C0h
MSCR97	3C4h
MSCR98	3C8h
MSCR99	3CCh
MSCR100	3D0h
MSCR101	3D4h
MSCR102	3D8h
MSCR103	3DCh
MSCR104	3E0h
MSCR105	3E4h
MSCR106	3E8h
MSCR107	3ECh
MSCR108	3F0h
MSCR109	3F4h
MSCR110	3F8h
MSCR111	3FCh
MSCR112	400h
MSCR113	404h
MSCR114	408h
MSCR115	40Ch
MSCR116	410h
MSCR117	414h
MSCR118	418h
MSCR119	41Ch
MSCR120	420h
MSCR121	424h
MSCR122	428h
MSCR123	42Ch
MSCR124	430h
MSCR125	434h
MSCR126	438h
MSCR127	43Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR128	440h
MSCR129	444h
MSCR130	448h
MSCR131	44Ch
MSCR132	450h
MSCR133	454h
MSCR134	458h
MSCR135	45Ch
MSCR136	460h
MSCR137	464h
MSCR138	468h
MSCR139	46Ch
MSCR140	470h
MSCR141	474h
MSCR142	478h
MSCR143	47Ch
MSCR144	480h
MSCR145	484h
MSCR146	488h
MSCR147	48Ch
MSCR148	490h
MSCR149	494h
MSCR150	498h
MSCR151	49Ch
MSCR152	4A0h
MSCR153	4A4h
MSCR154	4A8h
MSCR155	4ACh
MSCR156	4B0h
MSCR157	4B4h
MSCR158	4B8h
MSCR159	4BCh

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR160	4C0h
MSCR161	4C4h
MSCR162	4C8h
MSCR163	4CCh
MSCR164	4D0h
MSCR165	4D4h
MSCR166	4D8h
MSCR167	4DCh
MSCR168	4E0h
MSCR169	4E4h
MSCR170	4E8h
MSCR171	4ECh
MSCR172	4F0h
MSCR173	4F4h
MSCR174	4F8h
MSCR175	4FCh
MSCR176	500h
MSCR177	504h
MSCR178	508h
MSCR179	50Ch
MSCR180	510h
MSCR181	514h
MSCR182	518h
MSCR183	51Ch
MSCR184	520h
MSCR185	524h
MSCR186	528h
MSCR187	52Ch
MSCR188	530h
MSCR189	534h
MSCR190	538h
MSCR191	53Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR192	540h
MSCR193	544h
MSCR194	548h
MSCR195	54Ch
MSCR196	550h
MSCR197	554h
MSCR198	558h
MSCR199	55Ch
MSCR200	560h
MSCR201	564h
MSCR202	568h
MSCR203	56Ch
MSCR204	570h
MSCR205	574h
MSCR206	578h
MSCR207	57Ch
MSCR208	580h
MSCR209	584h
MSCR210	588h
MSCR211	58Ch
MSCR212	590h
MSCR213	594h
MSCR214	598h
MSCR215	59Ch
MSCR216	5A0h
MSCR217	5A4h
MSCR218	5A8h
MSCR219	5ACh

Function

Select the source signal connected to the register's associated destination, which is a chip output pin or a chip pin that can be configured as an output.

MSCR*n* also specifies the electrical properties of the associated pin.

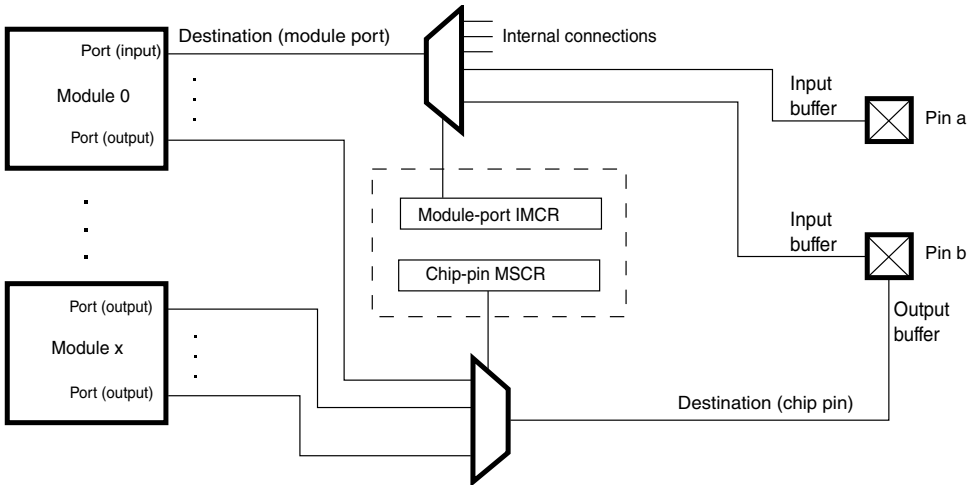


Figure 21. MSCR and IMCR port and pin connection

For chip-pin MSCR assignments and pin types, see the IOMUX file attached to this document.

NOTE

- MSCR*n* registers support only 32-bit accesses. Byte and half-word write accesses are not supported.
- MSCR*n* registers must be configured only during application initialization and must not be modified during application runtime.
- Accessing a reserved MSCR*n* register generates a transfer error.
- These registers are a part of SIUL memory map but physical implementation of these register is a part of IOMUX RTL.
- SIUL2 interprets accesses to MSCR*n* at module level.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0										OBE	0	IBE	0	INV	PKE
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SRC	PUE	0	PUS	0		DSE	0	IFE	SMC	0		SSS_2	SSS_1	SSS_0
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
MSCR0–MSCR3	0000_0000h
MSCR4	0008_2827h

Table continues on the next page...

Table continued from the previous page...

Register	Reset value
MSCR5–MSCR9	0000_0000h
MSCR10	0000_0127h
MSCR11	0000_0000h
MSCR12	0000_0003h
MSCR13–MSCR65	0000_0000h
MSCR38–MSCR39	Register not supported
MSCR66–MSCR67	0000_4000h
MSCR68	0008_2000h
MSCR69	0008_2800h
MSCR70–MSCR75	0000_0000h
MSCR76	0000_4000h
MSCR77–MSCR79	0000_0000h
MSCR80	0000_4000h
MSCR81–MSCR100	0000_0000h
MSCR101–MSCR103	0000_4000h
MSCR104–MSCR105	0000_0000h
MSCR106–MSCR108	0000_4000h
MSCR109–MSCR135	0000_0000h
MSCR136	0000_4000h
MSCR137–MSCR219	0000_0000h

Fields

Field	Function
31-22 —	Reserved
21 OBE	GPIO Output Buffer Enable Applies only to digital pins. Otherwise this bit is reserved. 0b - Output driver disabled 1b - Output driver enabled
20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 IBE	Input Buffer Enable Used only when the associated destination is a chip pin. Enables the associated pin's input buffer. 0b - Disabled 1b - Enabled
18 —	Reserved
17 INV	Invert Inverts the signal selected by SSS before transmitting it to the associated destination (chip pin or module port). 0b - Don't invert 1b - Invert
16 PKE	Pad keeping enable Pad keeping enable 0b - Disabled 1b - Enabled
15 —	Reserved
14 SRC	Slew Rate Control 0b - Fastest setting 1b - Slowest setting
13 PUE	Pull Enable Enables the pull function. Used only when the associated destination is a chip pin. 0b - Disabled 1b - Enabled
12 —	Reserved
11 PUS	Pull Select Determines whether the pull function is a pullup or pulldown when the pull function is enabled by the PUE field. Used only when the associated destination is a chip pin. 0b - Pull down 1b - Pull up

Table continues on the next page...

Table continued from the previous page...

Field	Function
10-9 —	Reserved
8 DSE	DSE Drive strength enable 0b - Disabled 1b - Enabled
7 —	Reserved
6 IFE	IFE Input filter enable **NOTE** This field is supported for RESET pad only (PTA5). 0b - Disabled 1b - Enabled
5 SMC	Safe Mode Control Used only when the associated destination is a chip pin. Specifies whether the chip disables the pin's output buffer when the chip enters Safe mode. 0b - Disable (The output buffer returns to its previous state when the chip leaves Safe mode.) 1b - Don't disable
4-3 —	Reserved
2 SSS_2	Source Signal Select_2 Selects a function for the pad. Refer to “SSS” column of the ‘IO Signal Table' tab of the IOMUX spreadsheet attachment.
1 SSS_1	Source Signal Select_1 Selects a function for the pad. Refer to “SSS” column of the ‘IO Signal Table' tab of the IOMUX spreadsheet attachment.
0 SSS_0	Source Signal Select_0 Selects a function for the pad. Refer to “SSS” column of the ‘IO Signal Table' tab of the IOMUX spreadsheet attachment.

9.6.14 SIUL2 Input Multiplexed Signal Configuration (IMCR0 - IMCR378)

Offset

Register	Offset
IMCR0	A40h
IMCR1	A44h
IMCR2	A48h
IMCR3	A4Ch
IMCR4	A50h
IMCR5	A54h
IMCR16	A80h
IMCR17	A84h
IMCR18	A88h
IMCR19	A8Ch
IMCR20	A90h
IMCR21	A94h
IMCR22	A98h
IMCR23	A9Ch
IMCR24	AA0h
IMCR25	AA4h
IMCR26	AA8h
IMCR27	AACh
IMCR28	AB0h
IMCR29	AB4h
IMCR30	AB8h
IMCR31	ABCh
IMCR32	AC0h
IMCR33	AC4h
IMCR34	AC8h
IMCR35	ACCh
IMCR36	AD0h
IMCR37	AD4h
IMCR38	AD8h
IMCR39	ADCh

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR40	AE0h
IMCR41	AE4h
IMCR42	AE8h
IMCR43	AECh
IMCR44	AF0h
IMCR45	AF4h
IMCR46	AF8h
IMCR47	AFCh
IMCR48	B00h
IMCR49	B04h
IMCR50	B08h
IMCR51	B0Ch
IMCR52	B10h
IMCR53	B14h
IMCR54	B18h
IMCR55	B1Ch
IMCR56	B20h
IMCR57	B24h
IMCR58	B28h
IMCR59	B2Ch
IMCR60	B30h
IMCR61	B34h
IMCR62	B38h
IMCR63	B3Ch
IMCR64	B40h
IMCR65	B44h
IMCR66	B48h
IMCR67	B4Ch
IMCR68	B50h
IMCR69	B54h
IMCR70	B58h
IMCR71	B5Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR80	B80h
IMCR81	B84h
IMCR82	B88h
IMCR83	B8Ch
IMCR84	B90h
IMCR85	B94h
IMCR86	B98h
IMCR87	B9Ch
IMCR88	BA0h
IMCR89	BA4h
IMCR90	BA8h
IMCR91	BACh
IMCR92	BB0h
IMCR93	BB4h
IMCR94	BB8h
IMCR95	BBCh
IMCR96	BC0h
IMCR97	BC4h
IMCR98	BC8h
IMCR99	BCCh
IMCR100	BD0h
IMCR101	BD4h
IMCR102	BD8h
IMCR103	BDCh
IMCR112	C00h
IMCR113	C04h
IMCR114	C08h
IMCR115	C0Ch
IMCR116	C10h
IMCR117	C14h
IMCR118	C18h
IMCR119	C1Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR120	C20h
IMCR121	C24h
IMCR122	C28h
IMCR123	C2Ch
IMCR124	C30h
IMCR125	C34h
IMCR126	C38h
IMCR127	C3Ch
IMCR128	C40h
IMCR129	C44h
IMCR130	C48h
IMCR131	C4Ch
IMCR132	C50h
IMCR133	C54h
IMCR134	C58h
IMCR135	C5Ch
IMCR144	C80h
IMCR145	C84h
IMCR146	C88h
IMCR147	C8Ch
IMCR148	C90h
IMCR149	C94h
IMCR152	CA0h
IMCR153	CA4h
IMCR154	CA8h
IMCR155	CACh
IMCR156	CB0h
IMCR157	CB4h
IMCR158	CB8h
IMCR159	CBCh
IMCR160	CC0h
IMCR161	CC4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR162	CC8h
IMCR163	CCCh
IMCR164	CD0h
IMCR165	CD4h
IMCR166	CD8h
IMCR167	CDCh
IMCR168	CE0h
IMCR169	CE4h
IMCR170	CE8h
IMCR171	CECh
IMCR172	CF0h
IMCR173	CF4h
IMCR174	CF8h
IMCR175	CFCh
IMCR176	D00h
IMCR177	D04h
IMCR178	D08h
IMCR179	D0Ch
IMCR180	D10h
IMCR181	D14h
IMCR182	D18h
IMCR183	D1Ch
IMCR184	D20h
IMCR185	D24h
IMCR186	D28h
IMCR187	D2Ch
IMCR188	D30h
IMCR189	D34h
IMCR190	D38h
IMCR191	D3Ch
IMCR192	D40h
IMCR193	D44h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR194	D48h
IMCR195	D4Ch
IMCR196	D50h
IMCR197	D54h
IMCR198	D58h
IMCR199	D5Ch
IMCR200	D60h
IMCR201	D64h
IMCR202	D68h
IMCR211	D8Ch
IMCR212	D90h
IMCR213	D94h
IMCR214	D98h
IMCR215	D9Ch
IMCR216	DA0h
IMCR217	DA4h
IMCR218	DA8h
IMCR219	DACh
IMCR220	DB0h
IMCR221	DB4h
IMCR222	DB8h
IMCR223	DBCh
IMCR224	DC0h
IMCR225	DC4h
IMCR226	DC8h
IMCR227	DCCh
IMCR228	DD0h
IMCR229	DD4h
IMCR230	DD8h
IMCR231	DDCh
IMCR232	DE0h
IMCR233	DE4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR234	DE8h
IMCR235	DECh
IMCR236	DF0h
IMCR237	DF4h
IMCR238	DF8h
IMCR239	DFCh
IMCR240	E00h
IMCR241	E04h
IMCR242	E08h
IMCR243	E0Ch
IMCR244	E10h
IMCR245	E14h
IMCR246	E18h
IMCR247	E1Ch
IMCR248	E20h
IMCR249	E24h
IMCR250	E28h
IMCR251	E2Ch
IMCR252	E30h
IMCR253	E34h
IMCR254	E38h
IMCR255	E3Ch
IMCR256	E40h
IMCR257	E44h
IMCR258	E48h
IMCR259	E4Ch
IMCR260	E50h
IMCR261	E54h
IMCR262	E58h
IMCR263	E5Ch
IMCR264	E60h
IMCR265	E64h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR266	E68h
IMCR267	E6Ch
IMCR268	E70h
IMCR289	EC4h
IMCR290	EC8h
IMCR291	ECCh
IMCR292	ED0h
IMCR293	ED4h
IMCR294	ED8h
IMCR295	EDCh
IMCR296	EE0h
IMCR297	EE4h
IMCR298	EE8h
IMCR299	EECh
IMCR300	EF0h
IMCR301	EF4h
IMCR302	EF8h
IMCR303	EFCh
IMCR304	F00h
IMCR305	F04h
IMCR306	F08h
IMCR307	F0Ch
IMCR308	F10h
IMCR309	F14h
IMCR315	F2Ch
IMCR316	F30h
IMCR317	F34h
IMCR318	F38h
IMCR319	F3Ch
IMCR320	F40h
IMCR321	F44h
IMCR322	F48h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR323	F4Ch
IMCR324	F50h
IMCR325	F54h
IMCR343	F9Ch
IMCR344	FA0h
IMCR345	FA4h
IMCR346	FA8h
IMCR347	FACh
IMCR348	FB0h
IMCR349	FB4h
IMCR350	FB8h
IMCR351	FBCh
IMCR352	FC0h
IMCR353	FC4h
IMCR354	FC8h
IMCR355	FCCh
IMCR356	FD0h
IMCR357	FD4h
IMCR358	FD8h
IMCR359	FDCh
IMCR360	FE0h
IMCR361	FE4h
IMCR362	FE8h
IMCR363	FECh
IMCR364	FF0h
IMCR365	FF4h
IMCR366	FF8h
IMCR367	FFCh
IMCR368	1000h
IMCR369	1004h
IMCR370	1008h
IMCR371	100Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR372	1010h
IMCR373	1014h
IMCR374	1018h
IMCR375	101Ch
IMCR376	1020h
IMCR377	1024h
IMCR378	1028h

Function

Select the source signal connected to the register's associated destination, which is an internal module port that is an input port or can be configured as an input.

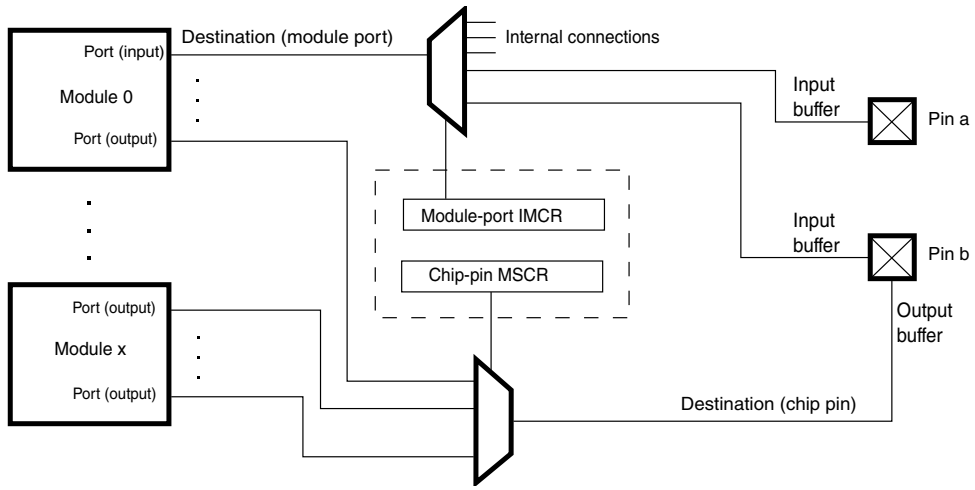


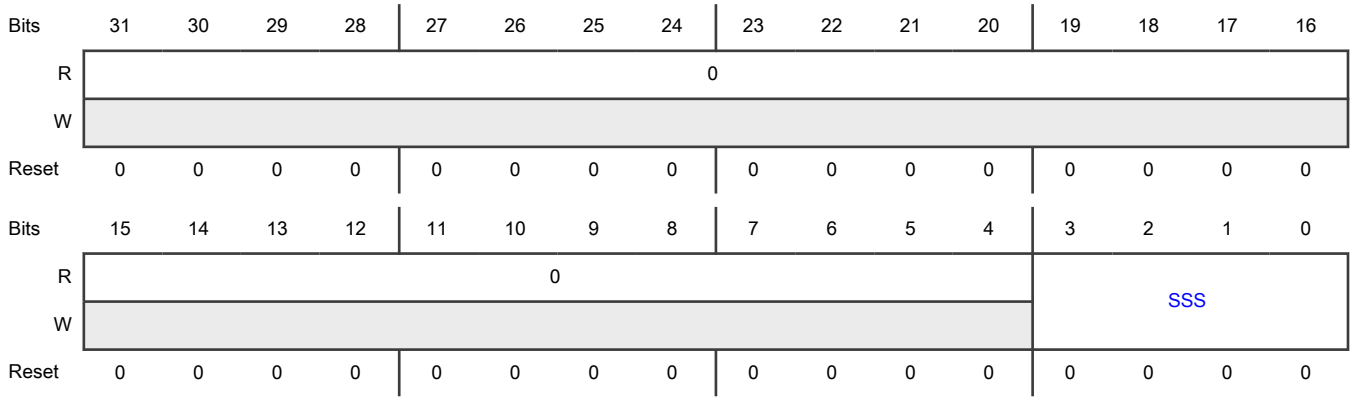
Figure 22. MSCR and IMCR port and pin connection

For IMCR assignments and field values, see the IOMUX file attached to this document.

NOTE

- IMCR*n* registers support only 32-bit accesses. Byte and half-word write accesses are not supported.
- IMCR*n* registers must be configured only during application initialization and must not be modified during application runtime.
- Accessing a reserved IMCR*n* register generates a transfer error.
- These registers are a part of SIUL memory map but physical implementation of these register is a part of IOMUX RTL.
- SIUL2 interprets accesses to MSCR*n* at module level.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 SSS	Source Signal Select Selects which source signal is connected to the associated destination (chip pin).

9.6.15 SIUL2 GPIO Pad Data Output (GPDO0 - GPDO219)

Offset

For n = 0 to 37; n = 40 to 219:

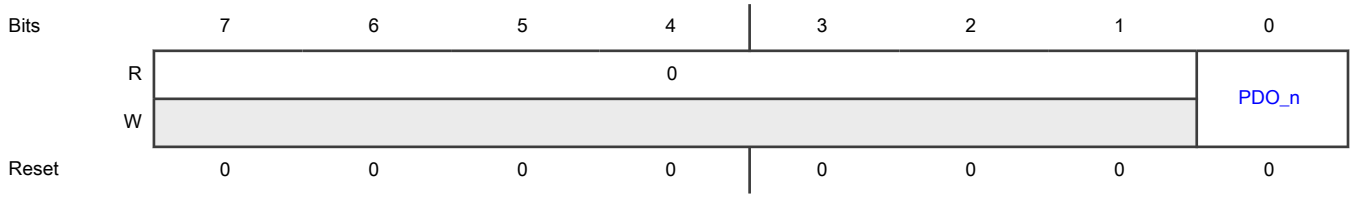
Register	Offset
GPDOn	1300h + (n + 3 - 2 × (n mod 4))

Function

Writes 0 or 1 to a single GPIO pad with a byte access.

These registers support 8-, 16-, and 32-bit accesses.

Diagram



Fields

Field	Function
7-1 —	Reserved
0 PDO_n	Pad Data Out Stores the data to be driven out on the external GPIO pad controlled by this register when the pad is configured as an output. PDO_*n* represents PDO[*n*], where *n* is the instance of the register. 0b - Pad Data Out Low. Logic low value 1b - Pad Data Out High. Logic high value

9.6.16 SIUL2 GPIO Pad Data Input (GPDI0 - GPDI219)

Offset

For n = 0 to 37; n = 40 to 219:

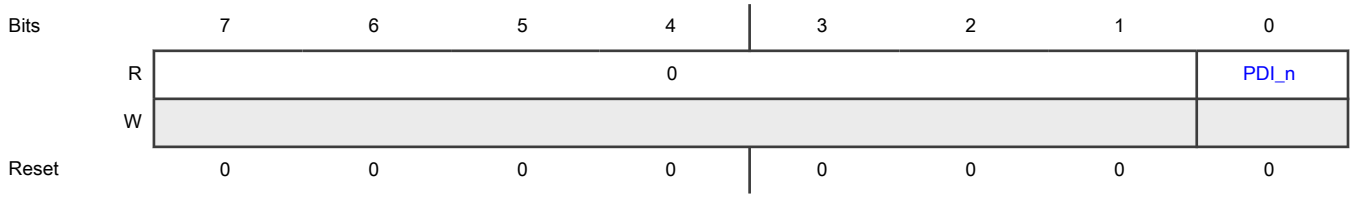
Register	Offset
GPDIn	1500h + (n + 3 - 2 × (n mod 4))

Function

Each GPDI*n* register reads the GPIO pad data with a byte access.

These registers support 8-, 16-, and 32-bit accesses.

Diagram



Fields

Field	Function
7-1 —	Reserved
0 PDI_n	Pad Data In Stores the value of the external GPIO pad associated with this register. PDI_*n* represents PDI[*n*], where *n* is the instance of the register. 0b - Pad Data In Low. Logic low 1b - Pad Data In High. Logic high

9.6.17 SIUL2 Parallel GPIO Pad Data Out (PGPDO0 - PGPDO3)

Offset

Register	Offset
PGPDO1	1700h
PGPDO0	1702h
PGPDO3	1704h

Function

Each PGPDO*n* register sets or clears the respective pads of the chip.

PGPDO*n* registers can set the values of all output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* registers set the value on a specific pin with byte writes.

Each bit writes or reads the data register that stores the value to be driven on the pad in output mode. Access to this register location is coherent with access to the bit-wise GPDO*n*.

For a given PGPDO*x*[PPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

PGPDO*n* registers access the same physical resource as the PDO and MPGPDO address locations.

These registers support 8-, 16-, and 32-bit accesses.

Diagram

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

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15 0b - Logic low 1b - Logic high
14 PPDO14	Parallel Pad Data Out 14 0b - Logic low 1b - Logic high
13 PPDO13	Parallel Pad Data Out 13 0b - Logic low 1b - Logic high
12 PPDO12	Parallel Pad Data Out 12 0b - Logic low 1b - Logic high
11 PPDO11	Parallel Pad Data Out 11 0b - Logic low 1b - Logic high
10 PPDO10	Parallel Pad Data Out 10 0b - Logic low 1b - Logic high
9 PPDO9	Parallel Pad Data Out 9 0b - Logic low 1b - Logic high
8 PPDO8	Parallel Pad Data Out 8 0b - Logic low 1b - Logic high

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 PPDO7	Parallel Pad Data Out 7 0b - Logic low 1b - Logic high
6 PPDO6	Parallel Pad Data Out 6 0b - Logic low 1b - Logic high
5 PPDO5	Parallel Pad Data Out 5 0b - Logic low 1b - Logic high
4 PPDO4	Parallel Pad Data Out 4 0b - Logic low 1b - Logic high
3 PPDO3	Parallel Pad Data Out 3 0b - Logic low 1b - Logic high
2 PPDO2	Parallel Pad Data Out 2 0b - Logic low 1b - Logic high
1 PPDO1	Parallel Pad Data Out 1 0b - Logic low 1b - Logic high
0 PPDO0	Parallel Pad Data Out 0 0b - Logic low 1b - Logic high

9.6.18 SIUL2 Parallel GPIO Pad Data Out (PGPDO2)

Offset

Register	Offset
PGPDO2	1706h

Function

Each PGPDO*n* register sets or clears the respective pads of the chip.

PGPDO*n* registers can set the values of all output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* registers set the value on a specific pin with byte writes.

Each bit writes or reads the data register that stores the value to be driven on the pad in output mode. Access to this register location is coherent with access to the bit-wise GPDO*n*.

For a given PGPDO*x*[PPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

PGPDO*n* registers access the same physical resource as the PDO and MPGPDO address locations.

These registers support 8-, 16-, and 32-bit accesses.

Diagram

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

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15 0b - Logic low 1b - Logic high
14 PPDO14	Parallel Pad Data Out 14 0b - Logic low 1b - Logic high
13 PPDO13	Parallel Pad Data Out 13 0b - Logic low 1b - Logic high
12 PPDO12	Parallel Pad Data Out 12 0b - Logic low 1b - Logic high
11 PPDO11	Parallel Pad Data Out 11 0b - Logic low

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Logic high
10 PPDO10	Parallel Pad Data Out 10 0b - Logic low 1b - Logic high
9 —	Reserved Always write zero to this field.
8 —	Reserved Always write zero to this field.
7 PPDO7	Parallel Pad Data Out 7 0b - Logic low 1b - Logic high
6 PPDO6	Parallel Pad Data Out 6 0b - Logic low 1b - Logic high
5 PPDO5	Parallel Pad Data Out 5 0b - Logic low 1b - Logic high
4 PPDO4	Parallel Pad Data Out 4 0b - Logic low 1b - Logic high
3 PPDO3	Parallel Pad Data Out 3 0b - Logic low 1b - Logic high
2 PPDO2	Parallel Pad Data Out 2 0b - Logic low 1b - Logic high
1 PPDO1	Parallel Pad Data Out 1 0b - Logic low 1b - Logic high
0 PPDO0	Parallel Pad Data Out 0 0b - Logic low

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Logic high

9.6.19 SIUL2 Parallel GPIO Pad Data Out (PGPDO4 - PGPDO11)

Offset

For n = 4 to 11:

Register	Offset
PGPDOn	1708h + 2 × (n + 1 – 2 × (n mod 2))

Function

Each PGPDO*n* register sets or clears the respective pads of the chip.

PGPDO*n* registers can set the values of all output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* registers set the value on a specific pin with byte writes.

Each bit writes or reads the data register that stores the value to be driven on the pad in output mode. Access to this register location is coherent with access to the bit-wise GPDO*n*.

For a given PGPDO*x*[PPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

PGPDO*n* registers access the same physical resource as the PDO and MPGPDO address locations.

These registers support 8-, 16-, and 32-bit accesses.

Diagram

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

Fields

Field	Function
15	Parallel Pad Data Out 15

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDO15	0b - Logic low 1b - Logic high
14 PPDO14	Parallel Pad Data Out 14 0b - Logic low 1b - Logic high
13 PPDO13	Parallel Pad Data Out 13 0b - Logic low 1b - Logic high
12 PPDO12	Parallel Pad Data Out 12 0b - Logic low 1b - Logic high
11 PPDO11	Parallel Pad Data Out 11 0b - Logic low 1b - Logic high
10 PPDO10	Parallel Pad Data Out 10 0b - Logic low 1b - Logic high
9 PPDO9	Parallel Pad Data Out 9 0b - Logic low 1b - Logic high
8 PPDO8	Parallel Pad Data Out 8 0b - Logic low 1b - Logic high
7 PPDO7	Parallel Pad Data Out 7 0b - Logic low 1b - Logic high
6 PPDO6	Parallel Pad Data Out 6 0b - Logic low 1b - Logic high
5 PPDO5	Parallel Pad Data Out 5 0b - Logic low 1b - Logic high

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 PPDO4	Parallel Pad Data Out 4 0b - Logic low 1b - Logic high
3 PPDO3	Parallel Pad Data Out 3 0b - Logic low 1b - Logic high
2 PPDO2	Parallel Pad Data Out 2 0b - Logic low 1b - Logic high
1 PPDO1	Parallel Pad Data Out 1 0b - Logic low 1b - Logic high
0 PPDO0	Parallel Pad Data Out 0 0b - Logic low 1b - Logic high

9.6.20 SIUL2 Parallel GPIO Pad Data Out (PGPDO13)

Offset

Register	Offset
PGPDO13	1718h

Function

Each PGPDO*n* register sets or clears the respective pads of the chip.

PGPDO*n* registers can set the values of all output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* registers set the value on a specific pin with byte writes.

Each bit writes or reads the data register that stores the value to be driven on the pad in output mode. Access to this register location is coherent with access to the bit-wise GPDO*n*.

For a given PGPDO*x*[PPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

PGPDO*n* registers access the same physical resource as the PDO and MPGPDO address locations.

These registers support 8-, 16-, and 32-bit accesses.

Diagram

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

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15 0b - Logic low 1b - Logic high
14 PPDO14	Parallel Pad Data Out 14 0b - Logic low 1b - Logic high
13 PPDO13	Parallel Pad Data Out 13 0b - Logic low 1b - Logic high
12 PPDO12	Parallel Pad Data Out 12 0b - Logic low 1b - Logic high
11 PPDO11	Parallel Pad Data Out 11 0b - Logic low 1b - Logic high
10 PPDO10	Parallel Pad Data Out 10 0b - Logic low 1b - Logic high
9 PPDO9	Parallel Pad Data Out 9 0b - Logic low 1b - Logic high
8 PPDO8	Parallel Pad Data Out 8 0b - Logic low 1b - Logic high

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 PPDO7	Parallel Pad Data Out 7 0b - Logic low 1b - Logic high
6 PPDO6	Parallel Pad Data Out 6 0b - Logic low 1b - Logic high
5 PPDO5	Parallel Pad Data Out 5 0b - Logic low 1b - Logic high
4 PPDO4	Parallel Pad Data Out 4 0b - Logic low 1b - Logic high
3 —	Reserved Always write zero to this field.
2 —	Reserved Always write zero to this field.
1 —	Reserved Always write zero to this field.
0 —	Reserved Always write zero to this field.

9.6.21 SIUL2 Parallel GPIO Pad Data Out (PGPDO12)

Offset

Register	Offset
PGPDO12	171Ah

Function

Each PGPDO*n* register sets or clears the respective pads of the chip.

PGPDO*n* registers can set the values of all output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* registers set the value on a specific pin with byte writes.

Each bit writes or reads the data register that stores the value to be driven on the pad in output mode. Access to this register location is coherent with access to the bit-wise GPDO*n*.

For a given PGPDO*x*[PPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

PGPDO*n* registers access the same physical resource as the PDO and MPGPDO address locations.

These registers support 8-, 16-, and 32-bit accesses.

Diagram

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

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15 0b - Logic low 1b - Logic high
14 PPDO14	Parallel Pad Data Out 14 0b - Logic low 1b - Logic high
13 PPDO13	Parallel Pad Data Out 13 0b - Logic low 1b - Logic high
12 PPDO12	Parallel Pad Data Out 12 0b - Logic low 1b - Logic high
11 PPDO11	Parallel Pad Data Out 11 0b - Logic low 1b - Logic high
10 PPDO10	Parallel Pad Data Out 10 0b - Logic low 1b - Logic high

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 PPDO9	Parallel Pad Data Out 9 0b - Logic low 1b - Logic high
8 PPDO8	Parallel Pad Data Out 8 0b - Logic low 1b - Logic high
7 PPDO7	Parallel Pad Data Out 7 0b - Logic low 1b - Logic high
6 PPDO6	Parallel Pad Data Out 6 0b - Logic low 1b - Logic high
5 PPDO5	Parallel Pad Data Out 5 0b - Logic low 1b - Logic high
4 PPDO4	Parallel Pad Data Out 4 0b - Logic low 1b - Logic high
3 PPDO3	Parallel Pad Data Out 3 0b - Logic low 1b - Logic high
2 PPDO2	Parallel Pad Data Out 2 0b - Logic low 1b - Logic high
1 PPDO1	Parallel Pad Data Out 1 0b - Logic low 1b - Logic high
0 PPDO0	Parallel Pad Data Out 0 0b - Logic low 1b - Logic high

9.6.22 SIUL2 Parallel GPIO Pad Data In (PGPDI0 - PGPDI3)

Offset

Register	Offset
PGPDI1	1740h
PGPDI0	1742h
PGPDI3	1744h

Function

Hold the synchronized input value from the pads.

PGPDI*n* registers can read the values of all input pins assigned to a chip port with a single 16-bit register read, while the GPDI*n* registers read the value on a specific pin with a byte read.

Each bit reads the current pad value. Access to this register location is coherent with access to the bit-wise GPDI*n*.

For a given PGPDI*x*[PPDI*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] bit:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	PPDI2	PPDI1	PPDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data In 15 0b - Logic low 1b - Logic high
14 PPDI14	Parallel Pad Data In 14 0b - Logic low

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Logic high
13 PPDI13	Parallel Pad Data In 13 0b - Logic low 1b - Logic high
12 PPDI12	Parallel Pad Data In 12 0b - Logic low 1b - Logic high
11 PPDI11	Parallel Pad Data In 11 0b - Logic low 1b - Logic high
10 PPDI10	Parallel Pad Data In 10 0b - Logic low 1b - Logic high
9 PPDI9	Parallel Pad Data In 9 0b - Logic low 1b - Logic high
8 PPDI8	Parallel Pad Data In 8 0b - Logic low 1b - Logic high
7 PPDI7	Parallel Pad Data In 7 0b - Logic low 1b - Logic high
6 PPDI6	Parallel Pad Data In 6 0b - Logic low 1b - Logic high
5 PPDI5	Parallel Pad Data In 5 0b - Logic low 1b - Logic high
4 PPDI4	Parallel Pad Data In 4 0b - Logic low 1b - Logic high

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 PPDI3	Parallel Pad Data In 3 0b - Logic low 1b - Logic high
2 PPDI2	Parallel Pad Data In 2 0b - Logic low 1b - Logic high
1 PPDI1	Parallel Pad Data In 1 0b - Logic low 1b - Logic high
0 PPDI0	Parallel Pad Data In 0 0b - Logic low 1b - Logic high

9.6.23 SIUL2 Parallel GPIO Pad Data In (PGPDI2)

Offset

Register	Offset
PGPDI2	1746h

Function

Hold the synchronized input value from the pads.

PGPDI*n* registers can read the values of all input pins assigned to a chip port with a single 16-bit register read, while the GPDI*n* registers read the value on a specific pin with a byte read.

Each bit reads the current pad value. Access to this register location is coherent with access to the bit-wise GPDI*n*.

For a given PGPDI*x*[PPDI*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] bit:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	Reserved	Reserved	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	PPDI2	PPDI1	PPDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data In 15 0b - Logic low 1b - Logic high
14 PPDI14	Parallel Pad Data In 14 0b - Logic low 1b - Logic high
13 PPDI13	Parallel Pad Data In 13 0b - Logic low 1b - Logic high
12 PPDI12	Parallel Pad Data In 12 0b - Logic low 1b - Logic high
11 PPDI11	Parallel Pad Data In 11 0b - Logic low 1b - Logic high
10 PPDI10	Parallel Pad Data In 10 0b - Logic low 1b - Logic high
9 —	Reserved
8 —	Reserved
7 PPDI7	Parallel Pad Data In 7 0b - Logic low 1b - Logic high

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 PPDI6	Parallel Pad Data In 6 0b - Logic low 1b - Logic high
5 PPDI5	Parallel Pad Data In 5 0b - Logic low 1b - Logic high
4 PPDI4	Parallel Pad Data In 4 0b - Logic low 1b - Logic high
3 PPDI3	Parallel Pad Data In 3 0b - Logic low 1b - Logic high
2 PPDI2	Parallel Pad Data In 2 0b - Logic low 1b - Logic high
1 PPDI1	Parallel Pad Data In 1 0b - Logic low 1b - Logic high
0 PPDI0	Parallel Pad Data In 0 0b - Logic low 1b - Logic high

9.6.24 SIUL2 Parallel GPIO Pad Data In (PGPDI4 - PGPDI11)

Offset

For n = 4 to 11:

Register	Offset
PGPDIn	1748h + 2 × (n + 1 – 2 × (n mod 2))

Function

Hold the synchronized input value from the pads.

PGPDI*n* registers can read the values of all input pins assigned to a chip port with a single 16-bit register read, while the GPDI*n* registers read the value on a specific pin with a byte read.

Each bit reads the current pad value. Access to this register location is coherent with access to the bit-wise GPDI*n*.

For a given PGPDI*x*[PPDI*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] bit:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	PPDI2	PPDI1	PPDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data In 15 0b - Logic low 1b - Logic high
14 PPDI14	Parallel Pad Data In 14 0b - Logic low 1b - Logic high
13 PPDI13	Parallel Pad Data In 13 0b - Logic low 1b - Logic high
12 PPDI12	Parallel Pad Data In 12 0b - Logic low 1b - Logic high
11 PPDI11	Parallel Pad Data In 11 0b - Logic low 1b - Logic high
10 PPDI10	Parallel Pad Data In 10 0b - Logic low

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Logic high
9 PPDI9	Parallel Pad Data In 9 0b - Logic low 1b - Logic high
8 PPDI8	Parallel Pad Data In 8 0b - Logic low 1b - Logic high
7 PPDI7	Parallel Pad Data In 7 0b - Logic low 1b - Logic high
6 PPDI6	Parallel Pad Data In 6 0b - Logic low 1b - Logic high
5 PPDI5	Parallel Pad Data In 5 0b - Logic low 1b - Logic high
4 PPDI4	Parallel Pad Data In 4 0b - Logic low 1b - Logic high
3 PPDI3	Parallel Pad Data In 3 0b - Logic low 1b - Logic high
2 PPDI2	Parallel Pad Data In 2 0b - Logic low 1b - Logic high
1 PPDI1	Parallel Pad Data In 1 0b - Logic low 1b - Logic high
0 PPDI0	Parallel Pad Data In 0 0b - Logic low 1b - Logic high

9.6.25 SIUL2 Parallel GPIO Pad Data In (PGPDI13)

Offset

Register	Offset
PGPDI13	1758h

Function

Hold the synchronized input value from the pads.

PGPDI*n* registers can read the values of all input pins assigned to a chip port with a single 16-bit register read, while the GPDI*n* registers read the value on a specific pin with a byte read.

Each bit reads the current pad value. Access to this register location is coherent with access to the bit-wise GPDI*n*.

For a given PGPDI*x*[PPDI*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] bit:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	PPDI4	Reserved	Reserved	Reserved	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data In 15 0b - Logic low 1b - Logic high
14 PPDI14	Parallel Pad Data In 14 0b - Logic low 1b - Logic high
13	Parallel Pad Data In 13

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDI13	0b - Logic low 1b - Logic high
12 PPDI12	Parallel Pad Data In 12 0b - Logic low 1b - Logic high
11 PPDI11	Parallel Pad Data In 11 0b - Logic low 1b - Logic high
10 PPDI10	Parallel Pad Data In 10 0b - Logic low 1b - Logic high
9 PPDI9	Parallel Pad Data In 9 0b - Logic low 1b - Logic high
8 PPDI8	Parallel Pad Data In 8 0b - Logic low 1b - Logic high
7 PPDI7	Parallel Pad Data In 7 0b - Logic low 1b - Logic high
6 PPDI6	Parallel Pad Data In 6 0b - Logic low 1b - Logic high
5 PPDI5	Parallel Pad Data In 5 0b - Logic low 1b - Logic high
4 PPDI4	Parallel Pad Data In 4 0b - Logic low 1b - Logic high
3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 —	Reserved
1 —	Reserved
0 —	Reserved

9.6.26 SIUL2 Parallel GPIO Pad Data In (PGPDI12)

Offset

Register	Offset
PGPDI12	175Ah

Function

Hold the synchronized input value from the pads.

PGPDI*n* registers can read the values of all input pins assigned to a chip port with a single 16-bit register read, while the GPDI*n* registers read the value on a specific pin with a byte read.

Each bit reads the current pad value. Access to this register location is coherent with access to the bit-wise GPDI*n*.

For a given PGPDI*x*[PPDI*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] bit:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	PPDI2	PPDI1	PPDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data In 15 0b - Logic low 1b - Logic high
14 PPDI14	Parallel Pad Data In 14 0b - Logic low 1b - Logic high
13 PPDI13	Parallel Pad Data In 13 0b - Logic low 1b - Logic high
12 PPDI12	Parallel Pad Data In 12 0b - Logic low 1b - Logic high
11 PPDI11	Parallel Pad Data In 11 0b - Logic low 1b - Logic high
10 PPDI10	Parallel Pad Data In 10 0b - Logic low 1b - Logic high
9 PPDI9	Parallel Pad Data In 9 0b - Logic low 1b - Logic high
8 PPDI8	Parallel Pad Data In 8 0b - Logic low 1b - Logic high
7 PPDI7	Parallel Pad Data In 7

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Logic low 1b - Logic high
6 PPDI6	Parallel Pad Data In 6 0b - Logic low 1b - Logic high
5 PPDI5	Parallel Pad Data In 5 0b - Logic low 1b - Logic high
4 PPDI4	Parallel Pad Data In 4 0b - Logic low 1b - Logic high
3 PPDI3	Parallel Pad Data In 3 0b - Logic low 1b - Logic high
2 PPDI2	Parallel Pad Data In 2 0b - Logic low 1b - Logic high
1 PPDI1	Parallel Pad Data In 1 0b - Logic low 1b - Logic high
0 PPDI0	Parallel Pad Data In 0 0b - Logic low 1b - Logic high

9.6.27 SIUL2 Masked Parallel GPIO Pad Data Out (MPGPDO0 - MPGPDO1)

Offset

Register	Offset
MPGPDO0	1780h
MPGPDO1	1784h

Function

Each MPGPDO*n* register selectively modifies the pad values associated with PGPDO*n*.

NOTE

MPGPDO*n* registers must only be accessed with 32-bit writes. 8-bit or 16-bit writes will not modify any bits in the register and cause a transfer error. Read access will return 0.

NOTE

MPGPDO*n* registers support only 32-bit accesses. Byte and half-word accesses are not supported.

Accesses to each MPGPDO*n* register location is coherent with access to the bit-wise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	MASK 9	MASK 8	MASK 7	MASK 6	MASK 5	MASK 4	MASK 3	MASK 2	MASK 1	MASK 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	MPPDO9	MPPDO8	MPPDO7	MPPDO6	MPPDO5	MPPDO4	MPPDO3	MPPDO2	MPPDO1	MPPDO0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MASK15	Mask Field 15 Masks MPPDO15 in the same MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the same MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29	Mask Field 13

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASK13	Masks MPPDO13 in the same MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the same MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the same MPGPDO*n* register instance. 0b - MPPDO11 is ignored 1b - MPPDO11 is written
26 MASK10	Mask Field 10 Masks MPPDO10 in the same MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 MASK9	Mask Field 9 Masks MPPDO9 in the same MPGPDO*n* register instance. 0b - MPPDO9 is ignored 1b - MPPDO9 is written
24 MASK8	Mask Field 8 Masks MPPDO8 in the same MPGPDO*n* register instance. 0b - MPPDO8 is ignored 1b - MPPDO8 is written
23 MASK7	Mask Field 7 Masks MPPDO7 in the same MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written
22 MASK6	Mask Field 6 Masks MPPDO6 in the same MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 MASK5	Mask Field 5 Masks MPPDO5 in the same MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written
20 MASK4	Mask Field 4 Masks MPPDO4 in the same MPGPDO*n* register instance. 0b - MPPDO4 is ignored 1b - MPPDO4 is written
19 MASK3	Mask Field 3 Masks MPPDO3 in the same MPGPDO*n* register instance. 0b - MPPDO3 is ignored 1b - MPPDO3 is written
18 MASK2	Mask Field 2 Masks MPPDO2 in the same MPGPDO*n* register instance. 0b - MPPDO2 is ignored 1b - MPPDO2 is written
17 MASK1	Mask Field 1 Masks MPPDO1 in the same MPGPDO*n* register instance. 0b - MPPDO1 is ignored 1b - MPPDO1 is written
16 MASK0	Mask Field 0 Masks MPPDO0 in the same MPGPDO*n* register instance. 0b - MPPDO0 is ignored 1b - MPPDO0 is written
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes the data register that stores the value to be driven on the pad in output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes the data register that stores the value to be driven on the pad in output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes the data register that stores the value to be driven on the pad in output mode.
12	Masked Parallel Pad Data Out 12

Table continues on the next page...

Table continued from the previous page...

Field	Function
MPPDO12	Writes the data register that stores the value to be driven on the pad in output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes the data register that stores the value to be driven on the pad in output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes the data register that stores the value to be driven on the pad in output mode.
9 MPPDO9	Masked Parallel Pad Data Out 9 Writes the data register that stores the value to be driven on the pad in output mode.
8 MPPDO8	Masked Parallel Pad Data Out 8 Writes the data register that stores the value to be driven on the pad in output mode.
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes the data register that stores the value to be driven on the pad in output mode.
6 MPPDO6	Masked Parallel Pad Data Out 6 Writes the data register that stores the value to be driven on the pad in output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes the data register that stores the value to be driven on the pad in output mode.
4 MPPDO4	Masked Parallel Pad Data Out 4 Writes the data register that stores the value to be driven on the pad in output mode.
3 MPPDO3	Masked Parallel Pad Data Out 3 Writes the data register that stores the value to be driven on the pad in output mode.
2 MPPDO2	Masked Parallel Pad Data Out 2 Writes the data register that stores the value to be driven on the pad in output mode.
1 MPPDO1	Masked Parallel Pad Data Out 1 Writes the data register that stores the value to be driven on the pad in output mode.
0 MPPDO0	Masked Parallel Pad Data Out 0 Writes the data register that stores the value to be driven on the pad in output mode.

9.6.28 SIUL2 Masked Parallel GPIO Pad Data Out (MPGPDO2)

Offset

Register	Offset
MPGPDO2	1788h

Function

Each MPGPDO*n* register selectively modifies the pad values associated with PGPDO*n*.

NOTE

MPGPDO*n* registers must only be accessed with 32-bit writes. 8-bit or 16-bit writes will not modify any bits in the register and cause a transfer error. Read access will return 0.

NOTE

MPGPDO*n* registers support only 32-bit accesses. Byte and half-word accesses are not supported.

Accesses to each MPGPDO*n* register location is coherent with access to the bit-wise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	Reserved	Reserved	MASK 7	MASK 6	MASK 5	MASK 4	MASK 3	MASK 2	MASK 1	MASK 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	Reserved	Reserved	MPPDO7	MPPDO6	MPPDO5	MPPDO4	MPPDO3	MPPDO2	MPPDO1	MPPDO0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Mask Field 15

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASK15	Masks MPPDO15 in the same MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the same MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29 MASK13	Mask Field 13 Masks MPPDO13 in the same MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the same MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the same MPGPDO*n* register instance. 0b - MPPDO11 is ignored 1b - MPPDO11 is written
26 MASK10	Mask Field 10 Masks MPPDO10 in the same MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 —	Reserved Always write zero to this field.
24 —	Reserved Always write zero to this field.
23 MASK7	Mask Field 7 Masks MPPDO7 in the same MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 MASK6	Mask Field 6 Masks MPPDO6 in the same MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written
21 MASK5	Mask Field 5 Masks MPPDO5 in the same MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written
20 MASK4	Mask Field 4 Masks MPPDO4 in the same MPGPDO*n* register instance. 0b - MPPDO4 is ignored 1b - MPPDO4 is written
19 MASK3	Mask Field 3 Masks MPPDO3 in the same MPGPDO*n* register instance. 0b - MPPDO3 is ignored 1b - MPPDO3 is written
18 MASK2	Mask Field 2 Masks MPPDO2 in the same MPGPDO*n* register instance. 0b - MPPDO2 is ignored 1b - MPPDO2 is written
17 MASK1	Mask Field 1 Masks MPPDO1 in the same MPGPDO*n* register instance. 0b - MPPDO1 is ignored 1b - MPPDO1 is written
16 MASK0	Mask Field 0 Masks MPPDO0 in the same MPGPDO*n* register instance. 0b - MPPDO0 is ignored 1b - MPPDO0 is written
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes the data register that stores the value to be driven on the pad in output mode.
14	Masked Parallel Pad Data Out 14

Table continues on the next page...

Table continued from the previous page...

Field	Function
MPPDO14	Writes the data register that stores the value to be driven on the pad in output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes the data register that stores the value to be driven on the pad in output mode.
12 MPPDO12	Masked Parallel Pad Data Out 12 Writes the data register that stores the value to be driven on the pad in output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes the data register that stores the value to be driven on the pad in output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes the data register that stores the value to be driven on the pad in output mode.
9 —	Reserved Always write zero to this field.
8 —	Reserved Always write zero to this field.
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes the data register that stores the value to be driven on the pad in output mode.
6 MPPDO6	Masked Parallel Pad Data Out 6 Writes the data register that stores the value to be driven on the pad in output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes the data register that stores the value to be driven on the pad in output mode.
4 MPPDO4	Masked Parallel Pad Data Out 4 Writes the data register that stores the value to be driven on the pad in output mode.
3 MPPDO3	Masked Parallel Pad Data Out 3 Writes the data register that stores the value to be driven on the pad in output mode.
2 MPPDO2	Masked Parallel Pad Data Out 2 Writes the data register that stores the value to be driven on the pad in output mode.
1 MPPDO1	Masked Parallel Pad Data Out 1 Writes the data register that stores the value to be driven on the pad in output mode.
0 MPPDO0	Masked Parallel Pad Data Out 0 Writes the data register that stores the value to be driven on the pad in output mode.

9.6.29 SIUL2 Masked Parallel GPIO Pad Data Out (MPGPDO3 - MPGPDO12)

Offset

For a = 3 to 12:

Register	Offset
MPGPDOa	1780h + (a × 4h)

Function

Each MPGPDO*n* register selectively modifies the pad values associated with PGPDO*n*.

NOTE

MPGPDO*n* registers must only be accessed with 32-bit writes. 8-bit or 16-bit writes will not modify any bits in the register and cause a transfer error. Read access will return 0.

NOTE

MPGPDO*n* registers support only 32-bit accesses. Byte and half-word accesses are not supported.

Accesses to each MPGPDO*n* register location is coherent with access to the bit-wise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	MASK 9	MASK 8	MASK 7	MASK 6	MASK 5	MASK 4	MASK 3	MASK 2	MASK 1	MASK 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	MPPDO9	MPPDO8	MPPDO7	MPPDO6	MPPDO5	MPPDO4	MPPDO3	MPPDO2	MPPDO1	MPPDO0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MASK15	Mask Field 15 Masks MPPDO15 in the same MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the same MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29 MASK13	Mask Field 13 Masks MPPDO13 in the same MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the same MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the same MPGPDO*n* register instance. 0b - MPPDO11 is ignored 1b - MPPDO11 is written
26 MASK10	Mask Field 10 Masks MPPDO10 in the same MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 MASK9	Mask Field 9 Masks MPPDO9 in the same MPGPDO*n* register instance. 0b - MPPDO9 is ignored 1b - MPPDO9 is written
24 MASK8	Mask Field 8 Masks MPPDO8 in the same MPGPDO*n* register instance. 0b - MPPDO8 is ignored

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - MPPDO8 is written
23 MASK7	Mask Field 7 Masks MPPDO7 in the same MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written
22 MASK6	Mask Field 6 Masks MPPDO6 in the same MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written
21 MASK5	Mask Field 5 Masks MPPDO5 in the same MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written
20 MASK4	Mask Field 4 Masks MPPDO4 in the same MPGPDO*n* register instance. 0b - MPPDO4 is ignored 1b - MPPDO4 is written
19 MASK3	Mask Field 3 Masks MPPDO3 in the same MPGPDO*n* register instance. 0b - MPPDO3 is ignored 1b - MPPDO3 is written
18 MASK2	Mask Field 2 Masks MPPDO2 in the same MPGPDO*n* register instance. 0b - MPPDO2 is ignored 1b - MPPDO2 is written
17 MASK1	Mask Field 1 Masks MPPDO1 in the same MPGPDO*n* register instance. 0b - MPPDO1 is ignored 1b - MPPDO1 is written
16 MASK0	Mask Field 0 Masks MPPDO0 in the same MPGPDO*n* register instance.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MPPDO0 is ignored 1b - MPPDO0 is written
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes the data register that stores the value to be driven on the pad in output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes the data register that stores the value to be driven on the pad in output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes the data register that stores the value to be driven on the pad in output mode.
12 MPPDO12	Masked Parallel Pad Data Out 12 Writes the data register that stores the value to be driven on the pad in output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes the data register that stores the value to be driven on the pad in output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes the data register that stores the value to be driven on the pad in output mode.
9 MPPDO9	Masked Parallel Pad Data Out 9 Writes the data register that stores the value to be driven on the pad in output mode.
8 MPPDO8	Masked Parallel Pad Data Out 8 Writes the data register that stores the value to be driven on the pad in output mode.
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes the data register that stores the value to be driven on the pad in output mode.
6 MPPDO6	Masked Parallel Pad Data Out 6 Writes the data register that stores the value to be driven on the pad in output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes the data register that stores the value to be driven on the pad in output mode.
4 MPPDO4	Masked Parallel Pad Data Out 4 Writes the data register that stores the value to be driven on the pad in output mode.
3 MPPDO3	Masked Parallel Pad Data Out 3 Writes the data register that stores the value to be driven on the pad in output mode.
2	Masked Parallel Pad Data Out 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
MPPDO2	Writes the data register that stores the value to be driven on the pad in output mode.
1 MPPDO1	Masked Parallel Pad Data Out 1 Writes the data register that stores the value to be driven on the pad in output mode.
0 MPPDO0	Masked Parallel Pad Data Out 0 Writes the data register that stores the value to be driven on the pad in output mode.

9.6.30 SIUL2 Masked Parallel GPIO Pad Data Out (MPGPDO13)

Offset

Register	Offset
MPGPDO13	17B4h

Function

Each MPGPDO*n* register selectively modifies the pad values associated with PGPDO*n*.

NOTE

MPGPDO*n* registers must only be accessed with 32-bit writes. 8-bit or 16-bit writes will not modify any bits in the register and cause a transfer error. Read access will return 0.

NOTE

MPGPDO*n* registers support only 32-bit accesses. Byte and half-word accesses are not supported.

Accesses to each MPGPDO*n* register location is coherent with access to the bit-wise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	MASK 9	MASK 8	MASK 7	MASK 6	MASK 5	MASK 4	Reserved	Reserved	Reserved	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	MPPDO9	MPPDO8	MPPDO7	MPPDO6	MPPDO5	MPPDO4	Reserved	Reserved	Reserved	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MASK15	Mask Field 15 Masks MPPDO15 in the same MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the same MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29 MASK13	Mask Field 13 Masks MPPDO13 in the same MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the same MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the same MPGPDO*n* register instance. 0b - MPPDO11 is ignored 1b - MPPDO11 is written
26	Mask Field 10

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASK10	Masks MPPDO10 in the same MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 MASK9	Mask Field 9 Masks MPPDO9 in the same MPGPDO*n* register instance. 0b - MPPDO9 is ignored 1b - MPPDO9 is written
24 MASK8	Mask Field 8 Masks MPPDO8 in the same MPGPDO*n* register instance. 0b - MPPDO8 is ignored 1b - MPPDO8 is written
23 MASK7	Mask Field 7 Masks MPPDO7 in the same MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written
22 MASK6	Mask Field 6 Masks MPPDO6 in the same MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written
21 MASK5	Mask Field 5 Masks MPPDO5 in the same MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written
20 MASK4	Mask Field 4 Masks MPPDO4 in the same MPGPDO*n* register instance. 0b - MPPDO4 is ignored 1b - MPPDO4 is written
19 —	Reserved Always write zero to this field.
18 —	Reserved Always write zero to this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 —	Reserved Always write zero to this field.
16 —	Reserved Always write zero to this field.
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes the data register that stores the value to be driven on the pad in output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes the data register that stores the value to be driven on the pad in output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes the data register that stores the value to be driven on the pad in output mode.
12 MPPDO12	Masked Parallel Pad Data Out 12 Writes the data register that stores the value to be driven on the pad in output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes the data register that stores the value to be driven on the pad in output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes the data register that stores the value to be driven on the pad in output mode.
9 MPPDO9	Masked Parallel Pad Data Out 9 Writes the data register that stores the value to be driven on the pad in output mode.
8 MPPDO8	Masked Parallel Pad Data Out 8 Writes the data register that stores the value to be driven on the pad in output mode.
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes the data register that stores the value to be driven on the pad in output mode.
6 MPPDO6	Masked Parallel Pad Data Out 6 Writes the data register that stores the value to be driven on the pad in output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes the data register that stores the value to be driven on the pad in output mode.
4 MPPDO4	Masked Parallel Pad Data Out 4 Writes the data register that stores the value to be driven on the pad in output mode.
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Always write zero to this field.
2 —	Reserved Always write zero to this field.
1 —	Reserved Always write zero to this field.
0 —	Reserved Always write zero to this field.

Chapter 10
Touch Sensing Pin Coupling (TSPC)

10.1 Chip-specific TSPC information

10.1.1 TSPC instances and configuration

This chip has one instance of TSPC module. The GPIO pads being used for TSPC should be configured with Source Signal Select (SSS) = 0. TSPC module works in conjunction with SIUL2.

NOTE

TSPC clock shall be enabled to control the I/Os assigned to TSPC.

For the GPIOs and ADC pads supporting TSPC, see the IOMUX file attached to this document. There are two groups of pads which can be configured via TSPC:

1. Group 1 consists of 32 ADC channels and 14 GPIOs
2. Group 2 consists of 32 ADC channels and 6 GPIOs

NOTE

Use preferably the same type of I/O pins for driving each electrode. Such I/O pins matching will improve resolution of the capacitance sensing. For more information about types of I/O pins, see the IOMUX file attached to this document.

10.2 Introduction

TSPC provides simultaneous transitioning of a group of pads into high impedance to support a more robust recognition of touch sense.

10.2.1 Block diagram

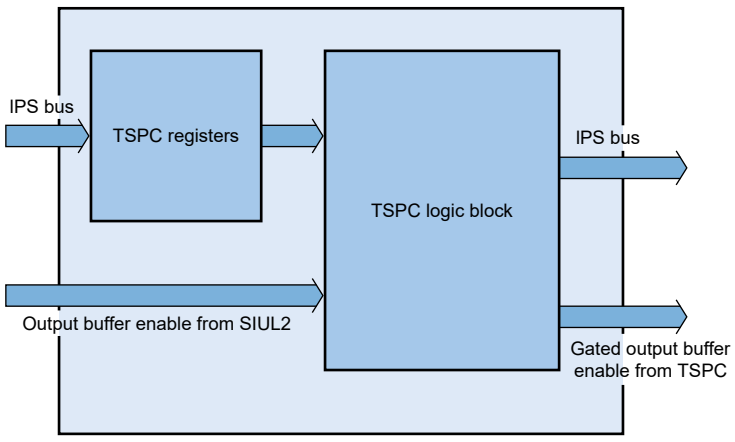


Figure 23. TSPC block diagram

10.2.2 Features

TSPC provides a set of control registers for pairing up GPIO pads for the touch sensing pairing operation. These control registers select the set of GPIO pads to pair with the set of ADC channels of an ADC instance. TSPC supports the following features:

- Up to 512 GPIO pads allow grouping

- OBE of all the grouped pads transitions from high to low simultaneously if SIUL2 transitions a pad's OBE from high to low
- Programming of group registers defines grouping

10.3 Logic block

This block takes the information from register fields to produce gated OBEs.

The simultaneous transition of grouped OBEs occurs when the GRP*n*_EN field in Group Enable (GRP_EN) for the respective group becomes 1.

The logic diagram below shows that if the GRP_EN is high for a particular group, and one of the output buffer enable (ipp_obe_in_b) from SIUL2 belonging to that group transitions from high to low, then all the other output buffer enables belonging to that group also transitions from high to low. If the GRP_EN is set low, the transitions in OBE do not affect the other OBE's belonging to that group.

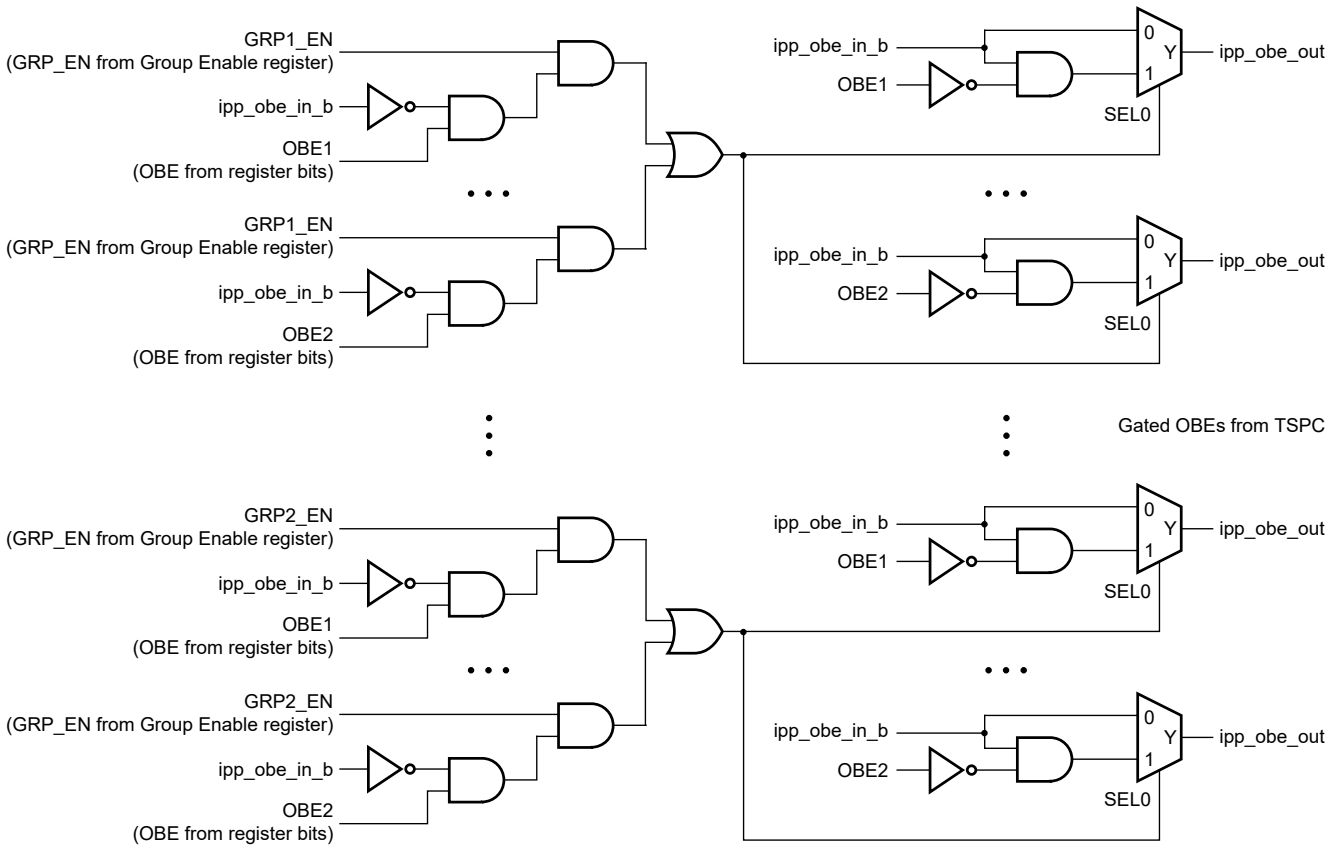


Figure 24. Logic block diagram

10.4 Functional description

TSPC contains these types of registers:

- Group enable: Each field of this register enables the respective group's OBE register—that is, if any OBE within a group transitions from high to low, it enables that group of OBEs for simultaneous transition from high to low. For example, GRP1_EN is the enable field for group 1 OBE registers. The number of fields in this register is equal to the number of groups.
- Group *n* OBE register *n*: The number of registers for a particular group *n* depends on the number of grouped OBEs in the respective group. For example, if the number of OBEs in group 1 is 36, then two registers of 32 fields each exist. Each field of this register represents an OBE pad. If you write 1 to the field, it indicates the OBE's participation in the grouping. If one of

the pad's OBEs transitions from high to low, all the participating OBEs transition from high to low. The remaining OBEs that are not participating in the grouping are not affected.

The register programming sequence is as follows:

1. Reset all the registers.
2. Write 1 to the GRP*n*_OBE*x*[OBE*n*] of the OBEs that participate in grouping.
3. Program the corresponding group enable field participating in grouping in Group Enable (GRP_EN).

You must write 0 to the corresponding GRP_EN field when you configure the group *n* OBE fields. You must write 1 to the OBE*n* field in Group OBE (GRP1_OBE1 - GRP2_OBE1) and then write 1 to the corresponding GRP_EN field in Group Enable (GRP_EN).

10.5 TSPC register descriptions

10.5.1 TSPC memory map

TSPC base address: 402C_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Group Enable (GRP_EN)	32	RW	0000_0000h
50h	Group OBE (GRP1_OBE1)	32	RW	0000_0000h
54h	Group OBE (GRP1_OBE2)	32	RW	0000_0000h
A0h	Group OBE (GRP2_OBE1)	32	RW	0000_0000h
A4h	Group OBE (GRP2_OBE2)	32	RW	0000_0000h

10.5.2 Group Enable (GRP_EN)

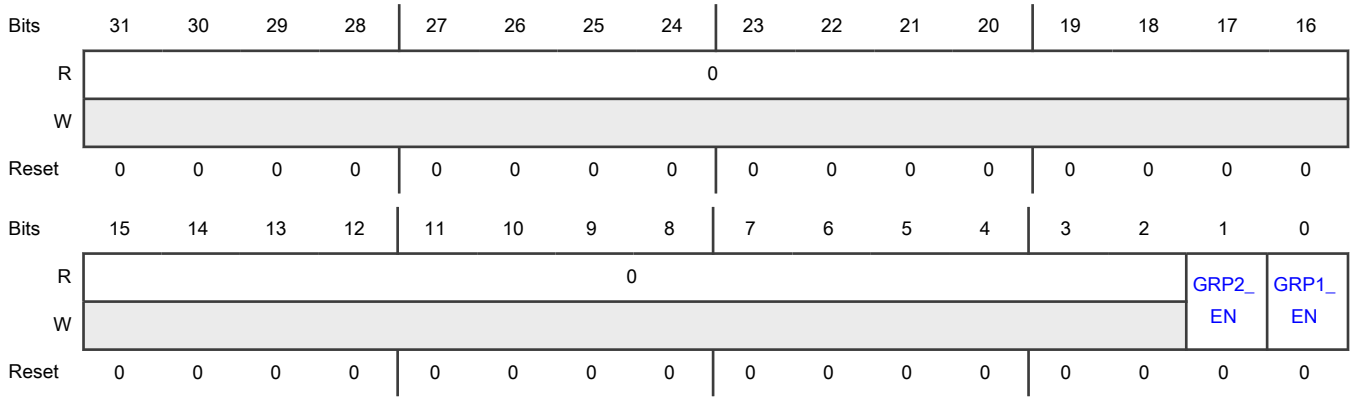
Offset

Register	Offset
GRP_EN	0h

Function

Allows enabling of the group, the pads of which participate in simultaneous transition.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 GRP2_EN	Enable for GRP2_OBEn Register Enables or disables the GRP2_OBE*n* register that participates in grouping. 0b - Disable 1b - Enable
0 GRP1_EN	Enable for GRP1_OBEn Register Enables or disables the GRP1_OBE*n* register that participates in grouping. 0b - Disable 1b - Enable

10.5.3 Group OBE (GRP1_OBE1 - GRP2_OBE1)

Offset

Register	Offset
GRP1_OBE1	50h
GRP2_OBE1	A0h

Function

Allows grouping of OBEs in the respective group. Each field corresponds to an OBE pad.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	OBE31	OBE30	OBE29	OBE28	OBE27	OBE26	OBE25	OBE24	OBE23	OBE22	OBE21	OBE20	OBE19	OBE18	OBE17	OBE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OBE15	OBE14	OBE13	OBE12	OBE11	OBE10	OBE9	OBE8	OBE7	OBE6	OBE5	OBE4	OBE3	OBE2	OBE1	OBE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OBEn	Output Buffer Enable Indicates if a particular OBE participates in grouping. 0b - OBE pad does not transition if any OBE in group transitions from high to low. The OBE pad value is unaltered at the output. 1b - OBE pad transitions from high to low if any OBE in group transitions from high to low.

10.5.4 Group OBE (GRP1_OBE2)

Offset

Register	Offset
GRP1_OBE2	54h

Function

Allows grouping of OBEs in the respective group. Each field corresponds to an OBE pad.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		OBE45	OBE44	OBE43	OBE42	OBE41	OBE40	OBE39	OBE38	OBE37	OBE36	OBE35	OBE34	OBE33	OBE32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-14 —	Reserved
13-0 OBEn	Output Buffer Enable Indicates if a particular OBE participates in grouping. 0b - OBE pad does not transition if any OBE in group transitions from high to low. The OBE pad value is unaltered at the output. 1b - OBE pad transitions from high to low if any OBE in group transitions from high to low.

10.5.5 Group OBE (GRP2_OBE2)

Offset

Register	Offset
GRP2_OBE2	A4h

Function
Allows grouping of OBEs in the respective group. Each field corresponds to an OBE pad.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										OBE37	OBE36	OBE35	OBE34	OBE33	OBE32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5-0	Output Buffer Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
OBEn	Indicates if a particular OBE participates in grouping. 0b - OBE pad does not transition if any OBE in group transitions from high to low. The OBE pad value is unaltered at the output. 1b - OBE pad transitions from high to low if any OBE in group transitions from high to low.

10.6 Glossary

GPIO	General purpose input and output
OBE	Output buffer enable

Chapter 11
Crossbar Switch (AXBS)

11.1 Chip-specific AXBS information

11.1.1 AXBS instances and connectivity matrix

This chip has up to seven instances of AXBS.

Table 35. AXBS instances

Instance	MWCT2D17S	MWCT2D16S	MWCT2016S	MWCT2015S
AXBS_0	Yes	Yes	Yes	Yes
AXBS_1	Yes	Yes	No	No
AXBS_2	Yes	Yes	No	No
AXBS_3	Yes	Yes	No	No
AXBS_4	Yes	Yes	No	No
AXBS_5	No	No	No	No
AXBS_6	No	No	No	No

Table 36. AXBS connectivity matrix for MWCT2D17S and MWCT2D16S

Instance	Master and slave assignments				
AXBS_0 (main)					
	Master port	**Master module**		**Slave port**	**Slave module**
	M0	Cortex-M7_0 AHBM		S0	Flash memory port 0
	M1	AXBS_2 S0		S1	Flash memory port 1
	M2	HSE_B		S2	PRAM_0
	M3	EMAC		S3	Cortex-M7 TCM
				S4	Flash memory port 2
				S5	QuadSPI
				S6	PRAM_1[1]
	1. These ports are reserved for MWCT2D16S.				
AXBS_1 (peripheral)					
	Master port	**Master module**		**Slave port**	**Slave module**
	M0	Cortex-M7_0 AHBP		S0	AIPS_0
	M1	AXBS_2 S1		S1	AIPS_1
	M2	HSE_B		S2	AIPS_2

Table 36. AXBS connectivity matrix for MWCT2D17S and MWCT2D16S (continued)

Instance	Master and slave assignments				
AXBS_2 (eDMA)					
	Master port	**Master module**		**Slave port**	**Slave module**
	M0	eDMA		S0	AXBS_0 M1
				S1	AXBS_1 M1
AXBS_3 (Cortex-M7 TCM)					
	Master port	**Master module**		**Slave port**	**Slave module**
	M0	AXBS_0 S3		S0	Cortex-M7_0 TCM
AXBS_4 (HSE)					
	Master port	**Master module**		**Slave port**	**Slave module**
	M0	HSE_B		S0	AXBS_0 M2

Table 37. AXBS connectivity matrix for MWCT2015S and MWCT2016S

Instance	Master and slave assignments				
AXBS_0 (main)					
	Master port	**Master module**		**Slave port**	**Slave module**
	M0	Cortex-M7_0 AHBM		S0	Flash memory port 0
	M1	AXBS_2 S0		S1	Flash memory port 1
	M2	HSE_B		S2	PRAM_0
	M3	EMAC		S3	Cortex-M7 TCM
	M4	Reserved		S4	AIPS_0
				S5	AIPS_1

11.2 Introduction

This section provides information on the layout, configuration, and programming of the crossbar switch.

The crossbar switch connects bus masters and bus slaves using a crossbar switch structure. This structure allows all bus masters to access different bus slaves simultaneously, while providing arbitration among the bus masters when they access the same slave. A variety of bus arbitration methods and attributes may be programmed on a slave-by-slave basis.

11.2.1 Features

- Symmetric crossbar bus switch implementation
  - Allows concurrent accesses from different masters to different slaves

— Slave arbitration attributes configured on a slave-by-slave basis

- Up to single-clock 64-bit transfer
- Support for burst transfers of up to 64 bits of data
- Support for low-power park mode
- Master high-priority elevation
- 64-bit AHB crossbar bus switch compatible with ARM's AMBA Specification v2.0

11.3 Functional description

Information about general operation and arbitration is provided in this section.

11.3.1 General operation

When a master accesses the crossbar switch, the access is immediately taken. If the targeted slave port of the access is available, then the access is immediately presented on the slave port. Single-clock or zero-wait-state accesses are possible through the crossbar. If the targeted slave port of the access is busy or parked on a different master port, the requesting master sees wait states inserted until the targeted slave port can service the master's request. The latency in servicing the request depends on each master's priority level and the responding slave's access time.

Because the crossbar switch appears to be just another slave to the master device, the master device has no knowledge of whether it actually owns the slave port it is targeting. The master waits while it does not have control of the slave port it is targeting.

After the master acquires control of the slave port, it controls the port until it relinquishes the port by running an IDLE cycle or by targeting a different slave port for its next access.

The master can also lose control of the slave port if another higher-priority master makes a request to the slave port. However, if the master is running a fixed-length burst transfer, it retains control of the slave port until that transfer completes.

The crossbar terminates all master IDLE transfers, as opposed to allowing the termination to come from one of the slave buses. Additionally, when no master is requesting access to a slave port, the crossbar drives IDLE transfers onto the slave bus, even though a default master may be granted access to the slave port.

When a slave bus is being idled by the crossbar, it can park the slave port on the master port indicated by CRS*n*[PARK]. This is done to save the initial clock of arbitration delay that otherwise would be seen if the same master had to arbitrate to gain control of the slave port. The slave port can also be put into low-power park mode to save power, by using CRS*n*[PCTL].

11.3.2 Register coherency

The operation of the crossbar is affected as soon as a register is written. The values of the registers do not track with slave-port-related master accesses, but instead track only with slave accesses.

11.3.3 Arbitration

The crossbar switch supports two arbitration algorithms:

- Fixed priority
- Round-robin

The arbitration scheme is independently programmable for each slave port.

11.3.3.1 Fixed-priority operation

When operating in fixed-priority mode, each master is assigned a unique priority level in the priority registers (PRS*n*). If two masters request access to the same slave port, the master with the highest priority in the selected priority register gains control over the slave port.

NOTE

In this arbitration mode, a higher-priority master can monopolize a slave port, preventing accesses from any lower-priority master to the port.

When a master makes a request to a slave port, the slave port checks whether the new requesting master's priority level is higher than that of the master that currently has control over the slave port, unless the slave port is in a parked state. The slave port performs an arbitration check at every clock edge to ensure that the master, if any, has control of the slave port.

The following table describes possible scenarios based on the requesting master port.

Table 38. Methods of how the crossbar switch grants control of a slave port to a master

When	Then the crossbar switch grants control to the requesting master
Both of the following are true: • The current master is not running a transfer. • The new requesting master's priority level is higher than that of the current master.	At the next clock edge
Both of the following are true: • The current master is running a fixed length burst transfer or a locked transfer. • The requesting master's priority level is higher than that of the current master.	At the end of the burst transfer or locked transfer
The requesting master's priority level is lower than the current master.	At the conclusion of one of the following cycles: • An IDLE cycle • A non-IDLE cycle to a location other than the current slave port

11.3.3.2 Round-robin priority operation

When operating in round-robin mode, each master is assigned a relative priority based on the master port number. This relative priority is compared to the master port number (ID) of the last master to perform a transfer on the slave bus. The highest priority requesting master owns the slave bus at the next transfer boundary, accounting for locked and fixed-length burst transfers. Priority is based on how far ahead the ID of the requesting master is to the ID of the last master.

After a master is granted access to a slave port, a master may perform as many transfers as desired to that port until another master makes a request to the same slave port. The next master in line is granted access to the slave port at the next transfer boundary, or possibly on the next clock cycle, if the current master has no pending access request.

As an example of arbitration in round-robin mode, assume that the crossbar is implemented with master ports 0, 1, 4, and 5. If the last master of the slave port was master 1, and masters 0, 4, and 5 make simultaneous requests, they are serviced in this order: 4,5, and then 0.

Parking may continue to be used in a round-robin mode, but does not affect the round-robin pointer unless the parked master performs a transfer. Handoff occurs to the next master in line after one cycle of arbitration. If the slave port is put into low-power park mode, the round-robin pointer is reset to point at master port 0, giving it the highest priority.

11.3.3.3 Priority assignment

Each master port must be assigned a unique 3-bit priority level. If an attempt is made to program multiple master ports with the same priority level within the priority registers (PRS*n*), the crossbar switch responds with a bus error and the registers are not updated.

11.4 Initialization/application information

No initialization is required for the crossbar switch.

Hardware reset ensures that all the register bits used by the crossbar switch are properly initialized to a valid state. However, the following settings and priorities may be programmed to achieve the maximum system performance:

- During the configuration of the crossbar switch, all other masters must be idle.
- To prevent reconfiguration of the crossbar switch, write 1 to CRS*n*[RO].

11.5 Memory map and register definition

Each slave port of the crossbar switch contains configuration registers. Read- and write-transfers require two bus clock cycles. The registers can be read from and written to only in supervisor mode. Additionally, these registers can be read from or written to only by 32-bit accesses.

A bus error response is returned if an unimplemented location is accessed within the crossbar switch.

The CRS*n* and PRS*n* registers can be programmed as read-only to prevent changes to their configuration. After being read-only protected, future writes to them terminate with a data storage error.

NOTE

This section shows the registers for all eight master and slave ports. If a master or slave is not used on this particular chip, then unexpected results occur when writing to its registers. See the chip configuration details for the exact master and slave assignments for your chip.

All references to the crossbar switch registers are based on the physical port connections.

11.5.1 AXBS register descriptions

11.5.1.1 AXBS memory map

AXBS_LITE base address: 4020_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Priority Slave Registers (PRS0)	32	RW	0004_3210h
10h	Control Register (CRS0)	32	RW	0002_0110h
100h	Priority Slave Registers (PRS1)	32	RW	0004_3210h
110h	Control Register (CRS1)	32	RW	0002_0110h
200h	Priority Slave Registers (PRS2)	32	RW	0004_3210h
210h	Control Register (CRS2)	32	RW	0002_0110h
300h	Priority Slave Registers (PRS3)	32	RW	0004_3210h
310h	Control Register (CRS3)	32	RW	0002_0110h
400h	Priority Slave Registers (PRS4)	32	RW	0004_3210h
410h	Control Register (CRS4)	32	RW	0002_0110h
500h	Priority Slave Registers (PRS5)	32	RW	0004_3210h
510h	Control Register (CRS5)	32	RW	0002_0110h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
600h	Priority Slave Registers (PRS6)	32	RW	0004_3210h
610h	Control Register (CRS6)	32	RW	0002_0110h

11.5.1.2 Priority Slave Registers (PRS0 - PRS6)

Offset

Register	Offset
PRS0	0h
PRS1	100h
PRS2	200h
PRS3	300h
PRS4	400h
PRS5	500h
PRS6	600h

Function

The priority slave registers(PRS*n*) set the priority of each master port on a per slave port basis and reside in each slave port. The priority register can be accessed only with 32-bit accesses. After the CRS*n*[RO] bit is set, the PRS*n* register can only be read; attempts to write to it have no effect on PRS*n* and result in a bus-error response to the master initiating the write.

Two available masters must not be programmed with the same priority level. Attempts to program two or more masters with the same priority level result in a bus-error response and the PRS*n* is not updated.

NOTE

Valid values for the M*n* priority fields depend on which masters are available on the chip. This information can be found in the chip-specific information for the crossbar.

- If the chip contains fewer than three masters, only one bit is valid.
- If the chip contains fewer than five masters, only two bits are valid.
- If five or more masters are present, all three bits of the priority field are used.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	Reserved			0	Reserved			0	Reserved			0	M4		
W		-				-				-						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	M3			0	M2			0	M1			0	M0		
W																
Reset	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 —	Reserved
27 —	Reserved
26-24 —	Reserved
23 —	Reserved
22-20 —	Reserved
19 —	Reserved
18-16 M4	Master 4 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or lowest priority when accessing the slave port.
15 —	Reserved
14-12 M3	Master 3 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or lowest priority when accessing the slave port.
11 —	Reserved
10-8 M2	Master 2 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or lowest priority when accessing the slave port.
7 —	Reserved
6-4	Master 1 Priority This field sets the arbitration priority for this port on the associated slave port.

Table continues on the next page...

Table continued from the previous page...

Field	Function
M1	000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or lowest priority when accessing the slave port.
3 —	Reserved
2-0 M0	Master 0 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or lowest priority when accessing the slave port.

11.5.1.3 Control Register (CRS0 - CRS6)

Offset

Register	Offset
CRS0	10h
CRS1	110h
CRS2	210h
CRS3	310h
CRS4	410h
CRS5	510h
CRS6	610h

Function

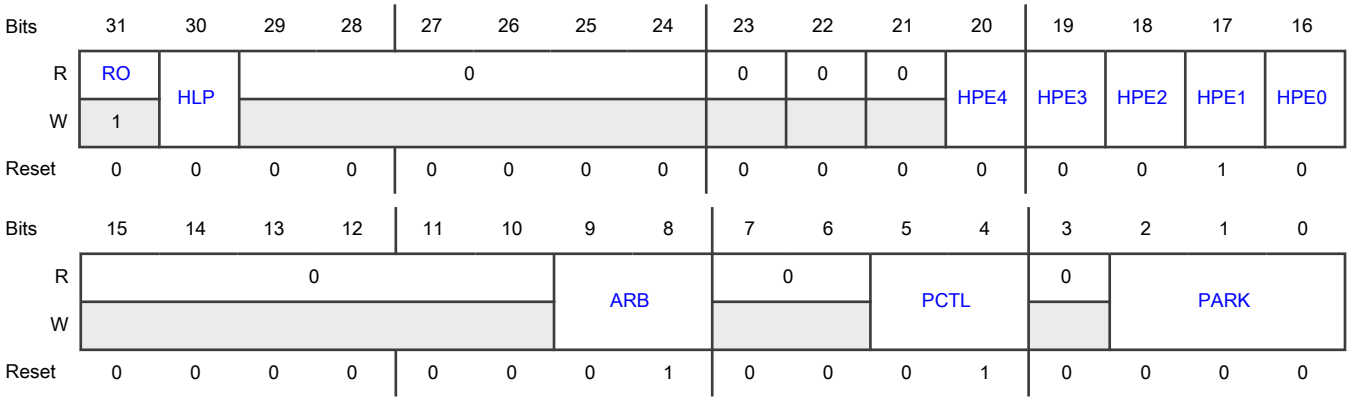
These registers control several features of each slave port and must be accessed using 32-bit accesses. After CRS*n*[RO] is set, the PRS*n* can only be read; attempts to write to it have no effect and result in an error response.

NOTE

See the chip-specific crossbar information for the reset value of this register.

Not all HPE *n* fields may be active. See the chip-specific crossbar information for which masters support master high-priority elevation. Setting a field corresponding to a master that does not support master high-priority elevation has no effect.

Diagram



Fields

Field	Function
31 RO	Read Only Forces the PRS*n* and CRS*n* registers to be read-only. After being set, only a hardware reset clears this field. 0b - The CRS*n* and PRS*n* registers are writeable 1b - The CRS*n* and PRS*n* registers are read-only and cannot be written (attempted writes have no effect on the registers and result in a bus error response).
30 HLP	Halt Low Priority This field sets the initial arbitration priority for low power-mode requests . Setting this bit will not affect the ,request for low power-mode from attaining highest priority after it has control of the slave ports. 0b - The low-power mode request has the highest priority for arbitration on this slave port. 1b - The low-power mode request has the lowest initial priority for arbitration on this slave port.
29-24 —	Reserved
23 —	Reserved
22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
21 —	Reserved
20 HPE4	High Priority Elevation 4 This field enables master high-priority elevation for master 4,on this slave port. If enabled, the master is able to elevate its priority to the highest. 0b - Master high-priority elevation for master 4. is disabled on this slave port. 1b - Master high-priority elevation for master 4. is enabled on this slave port.
19 HPE3	High Priority Elevation 3 This field enables master high-priority elevation for master 3,on this slave port. If enabled, the master is able to elevate its priority to the highest. 0b - Master high-priority elevation for master 3. is disabled on this slave port. 1b - Master high-priority elevation for master 3. is enabled on this slave port.
18 HPE2	High Priority Elevation 2 This field enables master high-priority elevation for master 2, on this slave port. If enabled, the master is able to elevate its priority to the highest. 0b - Master high-priority elevation for master 2. is disabled on this slave port. 1b - Master high-priority elevation for master 2. is enabled on this slave port.
17 HPE1	High Priority Elevation 1 This field enables master high-priority elevation for master 1 on this slave port. If enabled, the master is able to elevate its priority to the highest. 0b - Master high-priority elevation for master 1. is disabled on this slave port. 1b - Master high-priority elevation for master 1. is enabled on this slave port.
16 HPE0	High Priority Elevation 0 This field enables master high-priority elevation for master 0,on this slave port. If enabled, the master is able to elevate its priority to the highest. 0b - Master high-priority elevation for master 0. is disabled on this slave port. 1b - Master high-priority elevation for master 0. is enabled on this slave port.
15-10 —	Reserved
9-8 ARB	Arbitration Mode This field selects the arbitration policy for the slave port.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Fixed priority 01b - Round-robin (rotating) priority 10b - Reserved 11b - Reserved
7-6 —	Reserved
5-4 PCTL	Parking Control This field determines the slave port's parking control. The low-power park feature results in an overall power savings if the slave port is not saturated; however, this forces an extra latency clock when any master tries to access the slave port when not in use because it is not parked on any master. 00b - When no master makes a request, the arbiter parks the slave port on the master port defined by the PARK bit field. 01b - When no master makes a request, the arbiter parks the slave port on the last master to be in control of the slave port. 10b - Low-power park. When no master makes a request, the slave port is not parked on a master and the arbiter drives all outputs to a constant safe state. 11b - Reserved
3 —	Reserved
2-0 PARK	Park This field determines which master port the current slave port parks on when no masters are actively making requests and the PCTL bits are cleared. **NOTE** Select only master ports that are present on the chip. Otherwise, undefined behavior might occur. 000b - Park on master port M0 001b - Park on master port M1 010b - Park on master port M2 011b - Park on master port M3 100b - Park on master port M4 101b - Park on master port M5 110b - Park on master port M6 111b - Park on master port M7

11.6 Glossary

AMBA	Advanced Microcontroller Bus Architecture
IDLE	A type of transfer that a master uses when it does not want to perform a data transfer
ID	Master port number

Chapter 12
Peripheral Bridge (AIPS_Lite)

12.1 Chip-specific AIPS_Lite information

12.1.1 AIPS_Lite instances

The following table summarizes AIPS_Lite instances on MWCT2xxxS product series to support simultaneous data transfers to different peripherals. See the memory map file attached to this document to find information related to these peripherals that are associated with AIPS_Lite instances.

Table 39. AIPS_Lite instances

Instance	MWCT2D17S, MWCT2D16S	MWCT2016S, MWCT2015S
AIPS_0	Yes	Yes
AIPS_1	Yes	Yes
AIPS_2	Yes	No

12.2 Introduction

AIPS_Lite converts the crossbar switch interface to an interface that can access most of the slave peripherals on this chip.

This peripheral bridge occupies 64 MB of the address space, which is divided into peripheral slots of 16 KB each. All the peripherals may not be used. See the memory map chapter for details on slot assignments. The bridge includes separate clock enable inputs for each of the slots to accommodate slower peripherals.

12.2.1 Features

Following are the key features of the peripheral bridge:

- Supports peripheral slots with 8-, 16-, and 32-bit datapath width
- Supports a pair of 32-bit transactions for selected 64-bit memory accesses

12.2.2 General operation

The slave devices connected to the peripheral bridge are modules that contain a programming model of control and status registers. The system masters read and write these registers through the peripheral bridge.

The register maps of the peripherals are located on 16 KB boundaries. Each peripheral is allocated one or more 16-KB block(s) of the memory map.

12.3 Functional description

The peripheral bridge functions as a bus protocol translator between the crossbar switch and the slave peripheral bus. Support is provided for generating a pair of 32-bit slave accesses when performing certain 64-bit peripheral accesses.

The peripheral bridge manages all transactions for the attached slave devices and generates select signals for modules on the peripheral bus by decoding accesses within the attached address space.

12.3.1 Access support

All combinations of access size and peripheral data port width are supported. An access that is larger than the target peripheral's data width is decomposed to multiple, smaller accesses. Bus decomposition is terminated by a transfer error caused by an access to an empty register area.

12.4 Memory map and register definition

The AIPS module(s) on this chip do(es) not contain any user-programmable registers.

12.5 Glossary

ID	Master port number

Chapter 13
Direct Memory Access Multiplexer (DMAMUX)

13.1 Chip-specific DMAMUX information

13.1.1 DMAMUX instances

This chip has two instances of DMAMUX: DMAMUX_0 and DMAMUX_1. See the DMAMUX map file attached to this document to find information related to the slot number for each DMA trigger source.

NOTE

- For MWCT2014S, MWCT2015S, and MWCT2016S: DMAMUX_0 channels 0-5 and DMAMUX_1 channels 0-5 are mapped to eDMA Transfer Control Descriptor(TCD) 0-5 and eDMA Transfer Control Descriptor(TCD) 6-11, respectively. Programming of DMAMUX_0 channels 6-15 and DMAMUX_1 channels 6-15 is not therefore expected however if programmed then any access will terminate with either error response for channels 8-15 or without error response for channels 6-7.
- For remaining MWCT2xxxS devices: DMAMUX_0 channel 0-15 and DMAMUX_1 channel 0-15 are mapped to eDMA Transfer Control Descriptor(TCD) 0-15 and eDMA Transfer Control Descriptor(TCD) 16-31, respectively.

13.1.2 Periodic DMA triggering

PIT generates periodic trigger events to DMAMUX as shown in this table.

Table 40. PIT channel assignment for periodic DMA triggering through DMAMUX

DMAMUX0/1 ports	PIT channel
pit_dma_trigger[0]	PIT0 channel 0
pit_dma_trigger[1]	PIT0 channel 1
pit_dma_trigger[2]	PIT0 channel 2
pit_dma_trigger[3]	PIT0 channel 3

13.2 Introduction

13.2.1 Overview

DMAMUX routes a DMA source, called slot, to any of the 16 DMA channels. See the chip-specific information for details.

13.2.2 Block diagram

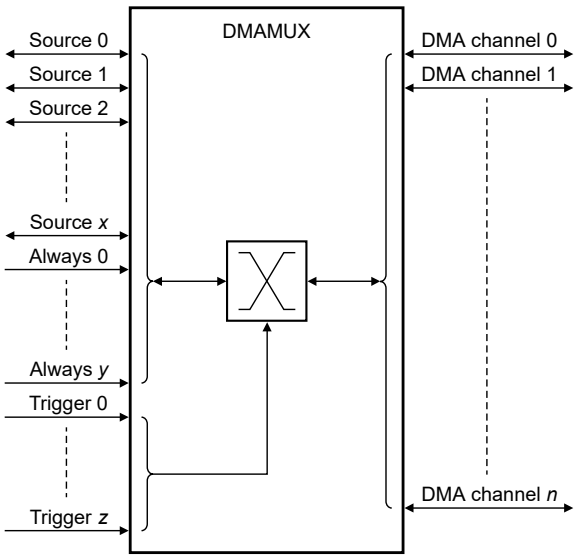


Figure 25. Block diagram

13.2.3 Features

- Ability to route up to 61 peripheral slots and up to 2 always-on slots to 16 channels
- Supports 16 independently selectable DMA channel routers (the first 4 channels provide an additional trigger functionality)
- Allows assignment of each channel router to one of the possible peripheral DMA slots or always-on slots

13.2.4 Modes of operation

The following table describes the functional operating modes of DMAMUX.

Table 41. Modes of operation

Mode	Description
Disabled	In this mode, the DMA channel is disabled. Because disabling and enabling of DMA channels is done primarily via the DMA configuration registers, this mode is used mainly as the reset state for a DMA channel in the DMA channel MUX. You can also use this mode to temporarily suspend a DMA channel while system reconfiguration takes place, for example, by changing the period of a DMA trigger.
Normal	In this mode, a DMA source is routed directly to the specified DMA channel. The operation of DMAMUX in this mode is completely transparent to the system.
Periodic trigger	In this mode, a DMA source can only request a DMA transfer, such as, when a transmit buffer becomes empty or a receive buffer becomes full, periodically. You configure the period using an external periodic interrupt timer (PIT). This mode is available only for channels 0–3. See the chip-specific section for more details.

13.3 Functional description

The primary purpose of DMAMUX is to provide flexibility in the system's use of the available DMA channels. Configuration of DMAMUX is intended to be a static procedure performed during the execution of the system boot code. However, if you follow the procedure outlined in Enabling and configuring sources, you can change the configuration of DMAMUX during the normal operation of the system. All DMA channels must be inactive before and during a configuration change.

Functionally, DMAMUX channels can be divided into two classes:

- Channels that implement the normal routing functionality plus the periodic triggering capability
- Channels that implement only the normal routing functionality

13.3.1 DMA channels with periodic triggering capability

Besides the normal routing functionality, the first four channels of DMAMUX provide a special periodic triggering capability that you can use to provide an automatic mechanism to transmit bytes, frames, or packets at fixed intervals without the need for processor intervention (see Figure 26).

An external periodic interrupt timer (PIT) generates the trigger; for example, by configuring the external periodic timer, you configure the periodic triggering interval.

NOTE

Because of the dynamic nature of the system (owing to DMA channel priorities, bus arbitration, interrupt service routine lengths, and so on), the number of clock cycles between a trigger and the actual DMA transfer cannot be guaranteed.

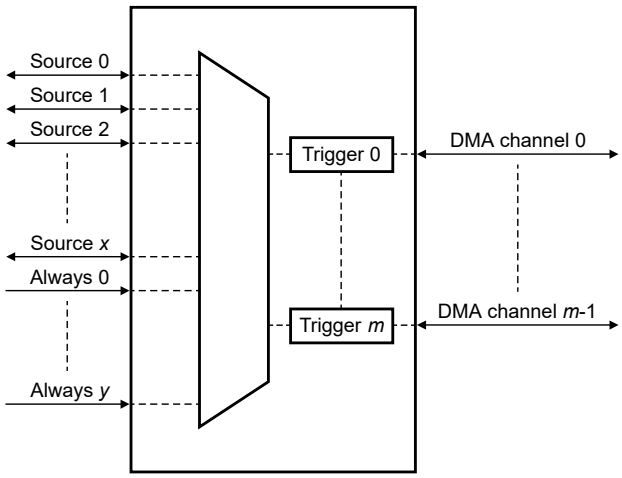


Figure 26. DMAMUX triggered channels

The DMA channel triggering capability allows the system to schedule regular DMA transfers, usually on the transmit side of certain peripherals, without the intervention of the processor. This trigger works by gating the request from the peripheral to DMA until a trigger event is observed (see Figure 27).

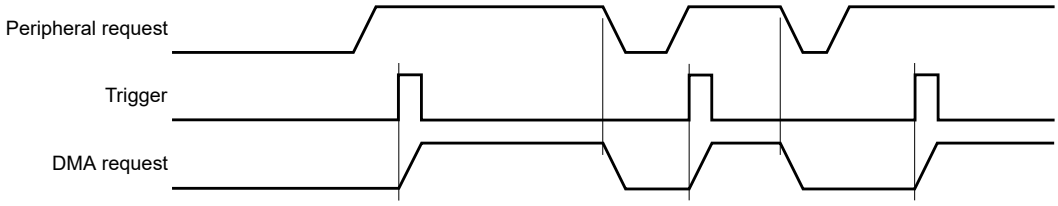


Figure 27. DMAMUX channel triggering: normal operation

After the DMA request is serviced, the peripheral negates its request, effectively resetting the gating mechanism until the peripheral reasserts its request and the next trigger event is seen. This means that if a trigger is seen, but the peripheral is not requesting a transfer, then that trigger is ignored (see Figure 28).

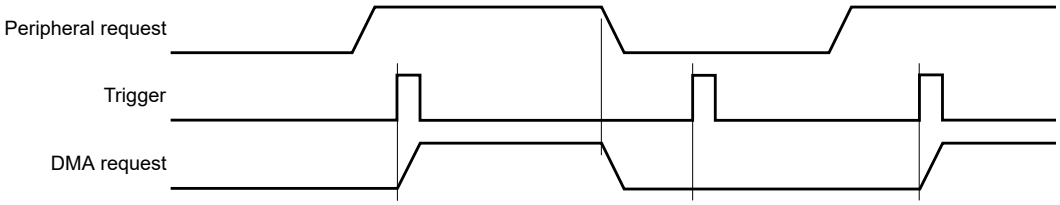


Figure 28. DMAMUX channel triggering: ignored trigger

You can use this triggering capability with any peripheral that supports DMA transfers, and is most useful in the following situations:

- Periodic polling of external devices on a particular bus: as an example, the transmit side of an SPI is assigned to a DMA channel with a trigger, as described in the aforementioned text. After periodic polling is set up, SPI requests DMA transfers, presumably from memory, as long as its transmit buffer is empty. By using a trigger on this channel, the SPI transfers can be automatically performed every 5 µs (as an example). On the receive side of SPI, you can configure SPI and DMA to transfer the received data into memory, effectively implementing a method to periodically read data from external devices and transferring the results into memory without processor intervention.
- Using the GPIO ports to drive or sample waveforms: by configuring DMA to transfer data to one or more GPIO ports, it is possible to create complex waveforms using tabular data stored in on-chip memory. Conversely, using DMA to periodically transfer data from one or more GPIO ports, it is possible to sample complex waveforms and store the results in a tabular form in on-chip memory.

13.3.2 DMA channels with no triggering capability

The other channels of DMAMUX provide the normal routing functionality as described in Modes of operation.

13.3.3 Always-enabled DMA sources

In addition to the peripherals that you can use as DMA sources, 2 additional DMA sources are always enabled. Unlike the peripheral DMA sources, where the peripheral controls the flow of data during DMA transfers, the sources that are always enabled provide no such "throttling" of data transfers. These sources are most useful in the following cases:

- Performing DMA transfers to and from GPIO: moving data to and from one or more GPIO pins, either unthrottled (that is, as fast as possible), or periodically (using the DMA triggering capability)
- Performing DMA transfers from memory to memory: moving data from memory to memory, typically as fast as possible, sometimes with software activation
- Performing DMA transfers from memory to the external bus, or vice-versa: similar to memory-to-memory transfers, this is typically done as quickly as possible
- Any DMA transfer that software must explicitly start or activate

In cases where software must initiate the start of a DMA transfer, an always-enabled DMA source can be used to provide maximum flexibility. When activating a DMA channel via software, subsequent executions of the minor loop require that a new start event be sent. This can either be a new software activation or a transfer request from the DMA channel MUX. The options for doing this are as follows.

- Transfer all data in a single minor loop: by configuring DMA to transfer all of the data in a single minor loop (that is, major loop counter = 1), no reactivation of the channel is necessary. The disadvantage of this option is reduced granularity in determining the load that the DMA transfer imposes on the system. For this option, the DMA channel must be disabled in the DMA channel MUX.
- Use explicit software reactivation: using this option, DMA is configured to transfer the data using both minor and major loops, but the processor is required to reactivate the channel by writing to the DMA registers after every minor loop. For this option, the DMA channel must be disabled in the DMA channel MUX.
- Use an always-enabled DMA source: using this option, DMA is configured to transfer the data using both minor and major loops, and the DMA channel MUX does the channel reactivation. For this option, you must enable the DMA channel and point it to an "always-enabled" source. Channel reactivation can be continuous (DMA triggering is disabled) or can use the DMA triggering capability. In this manner, it is possible to execute periodic transfers of data packets from one source to another, without processor intervention.

13.4 Initialization and application information

This section provides instructions for initializing the DMA channel MUX.

13.4.1 Reset

The reset state of each field is shown in Memory map and register definition. In summary, after reset, all channels are disabled and you must enable them explicitly before use.

13.4.2 Enabling and configuring sources

13.4.2.1 Enable a source with periodic triggering

Perform the following procedure to enable a source with periodic triggering:

1. Determine the DMA channel with which the source is associated. Only the first 4 DMA channels have periodic triggering capability.
2. Write 0 to CHCFG*n*[ENBL] and CHCFG*n*[TRIG] of the DMA channel.
3. Ensure that the DMA channel is properly configured in DMA. You can enable the DMA channel at this point.
4. Configure the corresponding timer.
5. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG*n*[ENBL] and CHCFG*n*[TRIG] are 1.

The following is an example. See the chip-specific information for the number of DMA channels on this chip that have triggering capability.

Example: To configure source 5 transmit for use with DMA channel 1, with periodic triggering capability:

1. Write 0h to CHCFG1.
2. Configure channel 1 in DMA, including enabling the channel.
3. Configure a timer for the desired trigger interval.
4. Write C5h to CHCFG1.

The following code example illustrates steps 1 and 4 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);

In File main.c:
#include "registers.h"
:
:
*CHCFG1 = 0x00;                              /* Clear all the fields of CHCFG1 register */
*CHCFG1 = 0xC5;                              /* ENBL = 1  DMA Channel is enabled */
                                             /* TRIG = 1  Triggering is enabled */
                                             /* SOURCE = 5  DMA Source 5 is selected */
```

Code Listing 1. Configuring source 5 with DMA channel 1 (with periodic triggering)

13.4.2.2 Enable a source without periodic triggering

Perform the following procedure to enable a source without periodic triggering:

1. Determine the DMA channel with which the source is associated. Only the first 4 DMA channels have periodic triggering capability.
2. Write 0 to CHCFG*n*[ENBL] and CHCFG*n*[TRIG] of the DMA channel.
3. Ensure that the DMA channel is properly configured in DMA. You can enable the DMA channel at this point.
4. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG*n*[ENBL] is 1 while CHCFG*n*[TRIG] is 0.

The following is an example. See the chip configuration details for the number of DMA channels on this chip that have triggering capability.

Example: To configure source 5 transmit for use with DMA channel 1, with no periodic triggering capability:

1. Write 0h to CHCFG1.
2. Configure channel 1 in DMA, including enabling the channel.
3. Write 85h to CHCFG1.

The following code example illustrates steps 1 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);

In File main.c:
#include "registers.h"
:
:
*CHCFG1 = 0x00;                              /* Clear all the fields of CHCFG1 register */
*CHCFG1 = 0x85;                              /* ENBL = 1  DMA Channel is enabled */
                                             /* TRIG = 0  Triggering is disabled */
                                             /* SOURCE = 5  DMA Source 5 is selected */
```

Code Listing 2. Configuring source 5 with DMA channel 1 (without periodic triggering)

13.4.2.3 Disable a source

You can disable a particular DMA source by not writing the corresponding source value into any of the CHCFG registers. Additionally, some module-specific configurations may be necessary. See the "Enhanced Direct Memory Access (eDMA)" chapter for details.

To switch the source of a DMA channel:

1. Disable the DMA channel in the DMA and reconfigure the channel for the new source.
2. Write 0 to CHCFG*n*[ENBL] and CHCFG*n*[TRIG] of the DMA channel.
3. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG*n*[ENBL] and CHCFG*n*[TRIG] are 1.

Example: To switch DMA channel 8 from source 5 transmit to source 7 transmit:

1. In the DMA configuration registers, disable DMA channel 8 and reconfigure it to handle the transfers to peripheral slot 7. This example assumes channel 8 does not have the triggering capability.
2. Write 0h to CHCFG8.
3. Write 87h to CHCFG8. (In this example, writing 1 to CHCFG[TRIG] has no effect because of the assumption that channel 8 does not support the periodic triggering functionality.)

The following code example illustrates steps 2 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);

In File main.c:
#include "registers.h"
:
:
*CHCFG8 = 0x00;                              /* Clear all the fields of CHCFG1 register */
*CHCFG8 = 0x87;                              /* ENBL = 1  DMA Channel is enabled */
                                             /* TRIG = 0  Triggering is disabled */
                                             /* SOURCE = 7  DMA Source 7 is selected */
```

Code Listing 3. Switching DMA channel 8 from source 5 transmit to source 7 transmit

13.5 Memory map and register definition

This section provides a detailed description of all memory-mapped registers in DMAMUX.

13.5.1 DMAMUX register descriptions

13.5.1.1 DMAMUX memory map

DMAMUX_0 base address: 4028_0000h

DMAMUX_1 base address: 4028_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Channel Configuration (CHCFG3)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1h	Channel Configuration (CHCFG2)	8	RW	00h
2h	Channel Configuration (CHCFG1)	8	RW	00h
3h	Channel Configuration (CHCFG0)	8	RW	00h
4h	Channel Configuration (CHCFG7)	8	RW	00h
5h	Channel Configuration (CHCFG6)	8	RW	00h
6h	Channel Configuration (CHCFG5)	8	RW	00h
7h	Channel Configuration (CHCFG4)	8	RW	00h
8h	Channel Configuration (CHCFG11)	8	RW	00h
9h	Channel Configuration (CHCFG10)	8	RW	00h
Ah	Channel Configuration (CHCFG9)	8	RW	00h
Bh	Channel Configuration (CHCFG8)	8	RW	00h
Ch	Channel Configuration (CHCFG15)	8	RW	00h
Dh	Channel Configuration (CHCFG14)	8	RW	00h
Eh	Channel Configuration (CHCFG13)	8	RW	00h
Fh	Channel Configuration (CHCFG12)	8	RW	00h

13.5.1.2 Channel Configuration (CHCFG0 - CHCFG15)

Offset

For n = 0 to 15:

Register	Offset
CHCFGn	0h + (n + 3 - 2 × (n mod 4))

Function

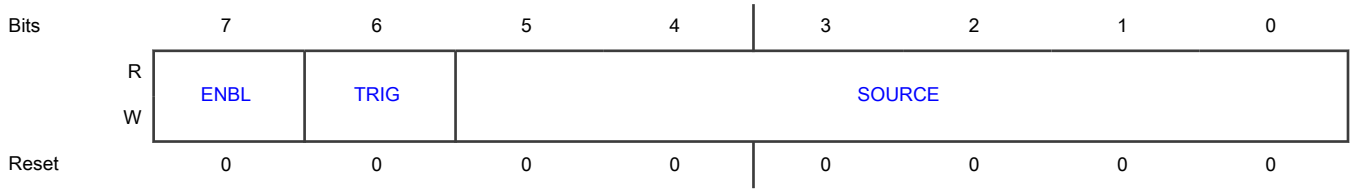
Provides you with the configuration to enable or disable, and to associate each of the DMA channels with one of the DMA slots (peripheral slots or always-on slots) in the system.

NOTE

Writing to multiple CHCFG registers with the same source value results in unpredictable behavior. This is true, even if a channel is disabled (CHCFG*n*[ENBL] == 0).

Before changing the trigger or source settings, you must disable a DMA channel using CHCFG*n*[ENBL].

Diagram



Fields

Field	Function
7 ENBL	DMA Channel Enable Enables the DMA channel. The condition when the DMA channel is disabled (ENBL = 0) is called Disabled mode. This mode is primarily used during the configuration of DMAMUX. DMA has separate channel enables or disables, which you must use to disable or reconfigure a DMA channel. 0b - Disable 1b - Enable
6 TRIG	DMA Channel Trigger Enable Enables the periodic triggering capability for the triggered DMA channel. If triggering is disabled and ENBL = 1, the DMA channel routes the specified source to the DMA channel (Normal mode). If triggering is enabled and ENBL = 1, then DMAMUX is in Periodic Trigger mode. 0b - Disable 1b - Enable
5-0 SOURCE	DMA Channel Source (Slot) Specifies which DMA source, if any, is routed to a particular DMA channel. See the chip-specific DMAMUX information for details about peripherals and their slot numbers.

13.6 Glossary

Slot	DMA source that either reflects a peripheral source or an always-enabled source
Periodic trigger	Trigger that an external source generates at periodic intervals for DMA transfer

Chapter 14
Enhanced Direct Memory Access (eDMA)

14.1 Chip-specific eDMA information

14.1.1 DMA channels

See the following table for number of DMA channels across MWCT2xxxS product series.

Table 42. DMA channels

Chip	No. of DMA channels
MWCT2D17S/MWCT2014S/MWCT2D16S	32
MWCT2015S/MWCT2016S	12

NOTE

TCD_CH[12-31] are not available in MWCT2014S, MWCT2015S and MWCT2016S, so the registers corresponding to these channels are not available. See the memory map file attached to this document for details.

14.1.2 eDMA ID replication

The eDMA replicates the IDs for the access based on the configurations of XRDC and eDMA itself. The below figure describes the details of the hmaster and DID corresponding the the eDMA.

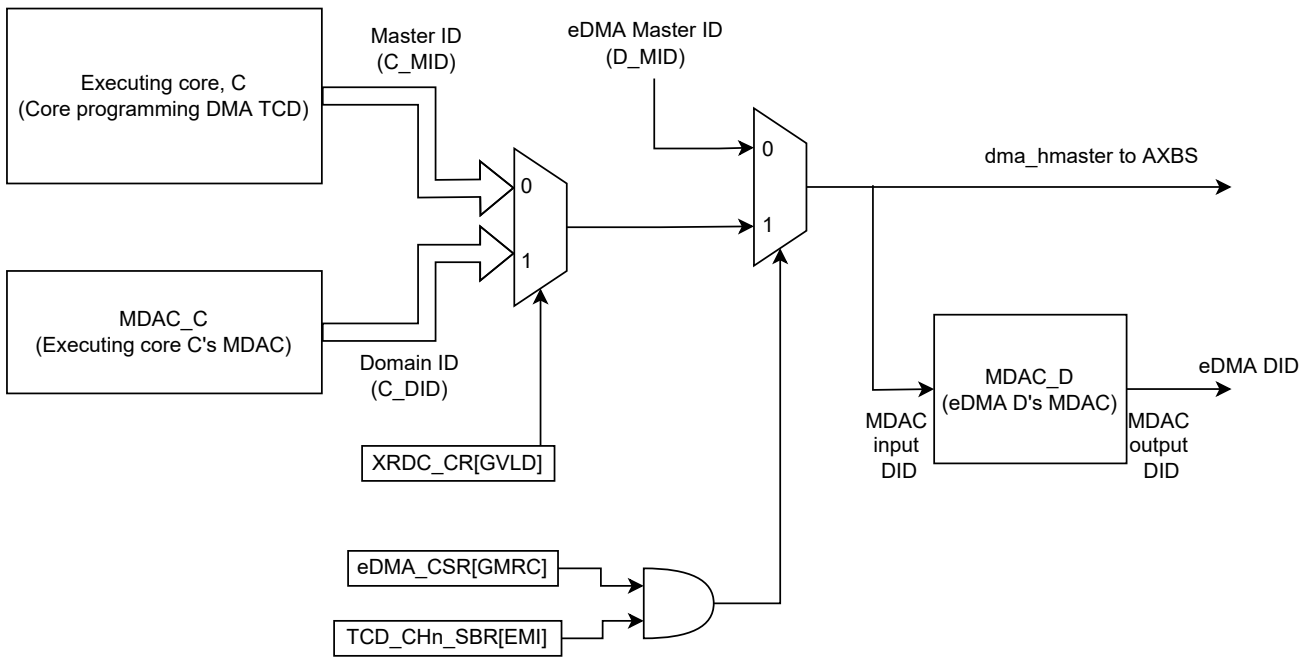


Figure 29. Details of the hmaster and DID corresponding to the eDMA

14.2 Overview

The enhanced direct memory access (eDMA) controller is capable of performing complex data transfers with minimal intervention from a host processor. The hardware microarchitecture includes:

- A DMA engine that performs:

— Source address and destination address calculations

— Data-movement operations

- Local memory containing transfer control descriptors for each of the 32 channels

14.2.1 Block diagram

Figure 30 illustrates the components of the eDMA system, including the eDMA module (engine).

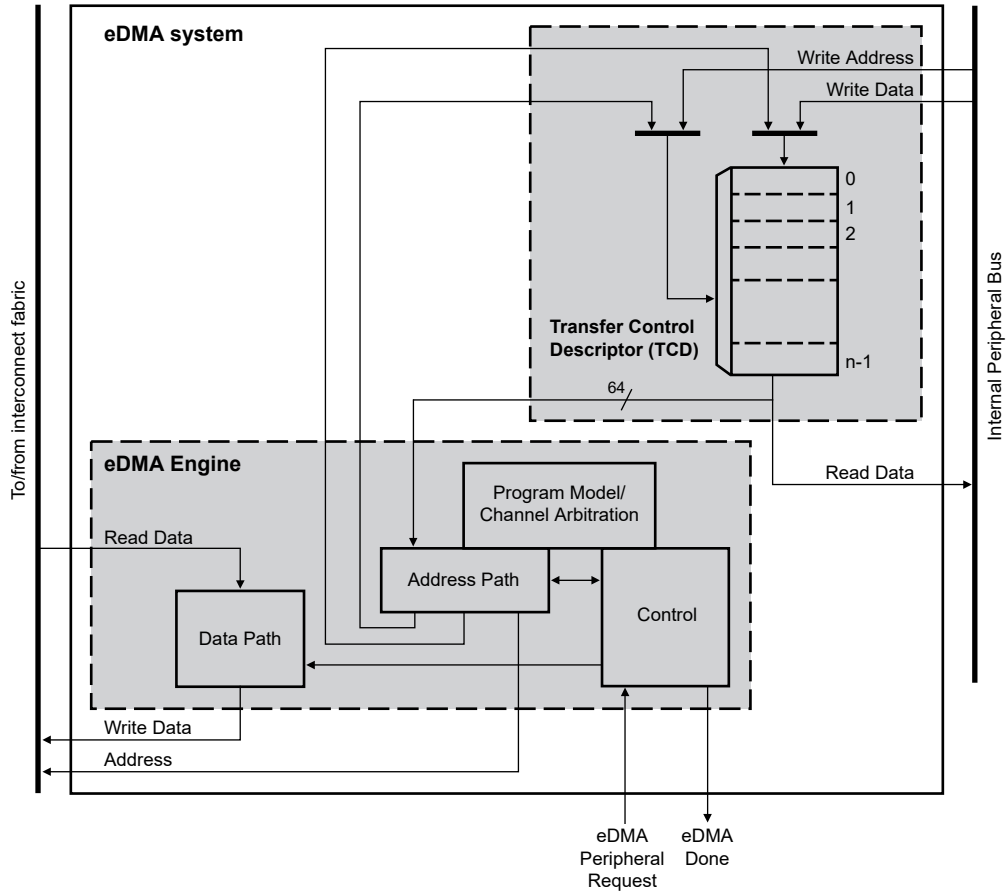


Figure 30. Block diagram

14.2.2 Block parts

The eDMA module is partitioned into two major modules: the eDMA engine and the transfer control descriptor local memory.

The eDMA engine is further partitioned into four submodules:

Table 43. eDMA engine submodules

Submodule	Function
Address path	This block: • Implements a primary channel and secondary (preempt) channel • Manages all master bus-address calculations

Table continues on the next page...

Table 43. eDMA engine submodules (continued)

Submodule	Function
	All the channels provide the same functionality. This structure allows data transfers associated with one channel to be preempted after the completion of a read/write sequence if a higher priority channel activation is asserted while the primary channel is active. After a channel is activated, it runs until the minor loop is completed, unless preempted by a higher priority channel. This provides a mechanism (enabled by CH*n*_PRI[ECP]) where a large data transfer can be preempted to minimize the time another channel is blocked from execution. When any channel is selected to execute, the contents of its TCD are read from local memory and loaded into the address path channel x registers for a normal start and into channel y registers for a preemption start. After the minor loop completes execution, the address path hardware writes the new values for the TCD*n*_{SADDR, DADDR, CITER} back to local memory. If the major iteration count is exhausted, additional processing is performed, including the final address pointer updates, reloading the TCD*n*_CITER field, and a possible fetch of a new TCD*n* from memory as part of a scatter/gather operation. See Dynamic scatter/gather for more details.
Data path	This block implements the bus master read/write data path. It includes a data buffer and the necessary multiplex logic to support any required data alignment. The internal read data bus is the primary input, and the internal write data bus is the primary output. The address and data path modules directly support the 2-stage pipelined internal bus. The address path module represents the first stage of the bus pipeline (address phase), and the data path module implements the second stage of the pipeline (data phase).
Program model/channel arbitration	This block implements the first section of the eDMA programming model as well as the channel arbitration logic. The programming model registers are connected to the internal peripheral bus. The eDMA peripheral request inputs and interrupt request outputs are also connected to this block (via control logic).
Control	This block provides all the control functions for the eDMA engine. For data transfers where the source and destination sizes are equal, the eDMA engine performs a series of source read/destination write operations until the number of bytes specified in the minor loop byte count has been moved from the source to the destination. For descriptors where the sizes are not equal, multiple accesses of the smaller size data are required for each reference of the larger size. As an example, if the source size references 16-bit data and the destination is 32-bit data, the eDMA performs two reads, then one 32-bit write.

The transfer control descriptor local memory is further partitioned into:

Table 44. Transfer control descriptor memory

Submodule	Description
Memory controller	This logic implements the required dual-ported controller, and manages accesses from the eDMA engine as well as references from the internal peripheral bus. In simultaneous accesses, the eDMA engine is given priority and the peripheral transaction is stalled.
Memory array	TCD storage for each channel's transfer profile.

14.2.3 Features

The eDMA is a highly programmable data-transfer engine optimized to minimize any required intervention from the host processor. It is intended for use in applications where the data size to be transferred is statically known and is not defined within the transferred data itself. The eDMA module features:

- All data movement via dual-address transfers: read from source, write to destination
  - Programmable source and destination addresses and transfer size
  - Support for complex address calculations
- 32-channel implementation that performs complex data transfers with minimal intervention from a host processor
  - Internal data buffer, used as temporary storage for all transfers
  - Connections to the crossbar switch for bus mastering the data movement
- TCD organized to support two-deep, nested transfer operations
  - 32-byte TCD stored in local memory for each channel
  - An inner data transfer loop defined by a minor byte transfer count
  - An outer data transfer loop defined by a major iteration count
- Channel activation via one of three methods:
  - Explicit software initiation
  - Initiation via a channel-to-channel linking mechanism for continuous transfers
  - Peripheral-paced hardware requests, one per channel
- Fixed-priority and round-robin channel arbitration
- Channel completion reported via programmable interrupt requests
  - One interrupt per channel, which can be asserted at completion of major iteration count
  - Programmable error terminations per channel that are logically summed together to form one error interrupt to the interrupt controller
- Programmable support for scatter/gather DMA processing
- Support for complex data structures

In the discussion of this module, *n* is used to reference the channel number.

14.3 Functional description

The operation of eDMA is described in the following subsections.

14.3.1 Modes of operation

The eDMA operates in the following modes:

Table 45. Modes of operation

Mode	Description
Normal	In Normal mode, eDMA transfers data between a source and a destination. The source and destination can be a memory block or an I/O block capable of operation with eDMA. A service request initiates a transfer of a specific number of bytes (NBYTES) as specified in the TCD. The minor loop is the sequence of read-write operations that transfers these NBYTES per service request. Each service request executes one iteration of the major loop, which transfers NBYTES of data.
Debug	eDMA operation is configurable in Debug mode via the control register: • If CSR[EDBG] is cleared to 0, eDMA continues to operate. • If CSR[EDBG] is set to 1, eDMA stops transferring data. If Debug mode is entered when a channel is active, eDMA continues operation until the channel retires.

14.3.2 eDMA basic data flow

The basic flow of a data transfer can be partitioned into three segments.

As shown in the following diagram, the first segment involves the channel activation:

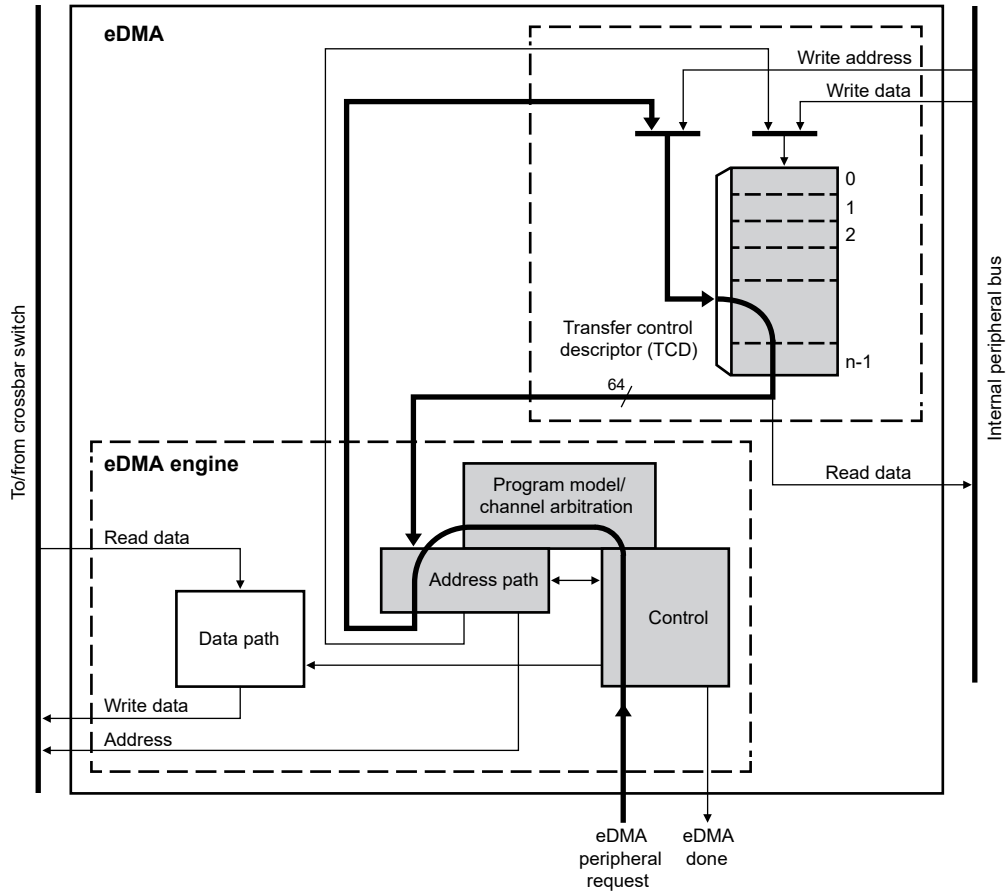


Figure 31. eDMA operation, part 1

This example uses the assertion of the eDMA peripheral request signal to request service for channel *n*. Channel activation via software and the TCD*n*_CSR[START] field follows the same basic flow as peripheral requests. The eDMA request input signal is registered internally and then routed through the eDMA engine: first through the control module, then into the program model and channel arbitration.

In the next cycle, the channel arbitration begins using fixed-priority plus the optional round-robin algorithm. After arbitration is complete, the activated channel number is sent through the address path and converted into the required address to access the local memory for TCD*n*. Next, the TCD memory is accessed and the required descriptor is read from the local memory and then loaded into the eDMA engine address path's primary or secondary channel execution registers. The TCD memory is 64 bits wide to minimize the time needed to fetch the activated channel descriptor and load it into the address path registers.

The following diagram illustrates the second part of the basic data flow:

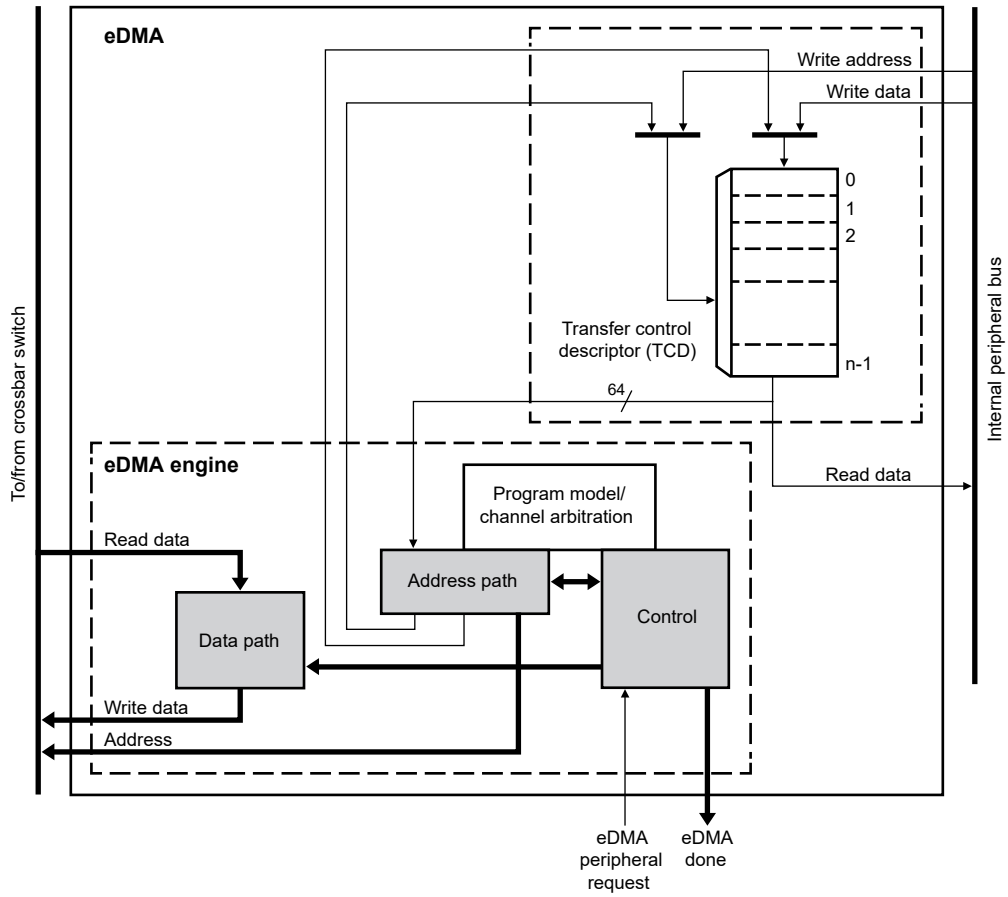


Figure 32. eDMA operation, part 2

The modules associated with the data transfer (address path, data path, and control) go through the required sequence of source reads and destination writes to perform the actual data movement. The source reads are initiated, and the fetched data is temporarily stored in the data path block until it is gated onto the internal bus during the destination write. This source read/destination write processing continues until the byte count, NBYTES, has been transferred.

After NBYTES of data has been moved, the final phase of the basic data flow is performed. In this segment, the address path logic performs the required updates to certain fields in the appropriate TCD (for example, SADDR, DADDR, CITER). If the major iteration count is exhausted, additional operations are performed. These include the final address adjustments and reloading of the BITER field into the CITER field. Assertion of an optional interrupt request also occurs at this time, as does a possible fetch of a new TCD from memory using the scatter/gather address pointer included in the descriptor (if scatter/gather is enabled). The updates to the TCD memory and the assertion of an interrupt request are shown in the following diagram.

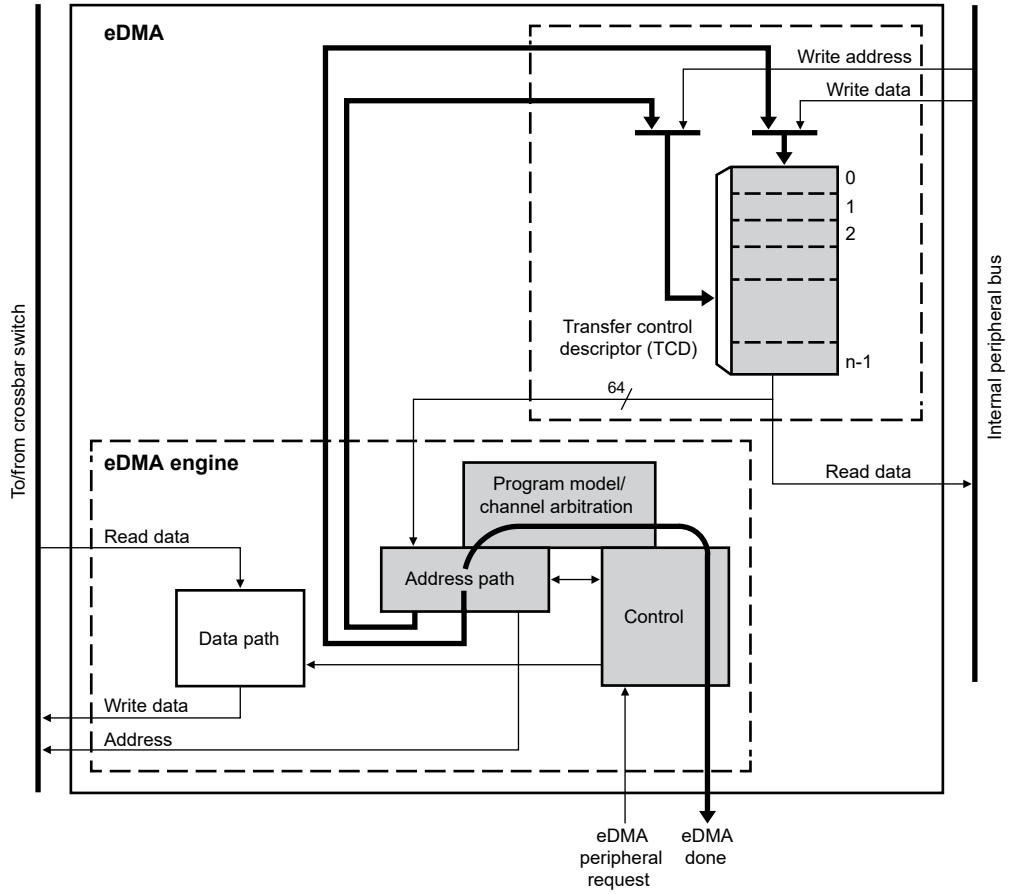


Figure 33. eDMA operation, part 3

14.3.3 Fault reporting and handling

Channel errors are reported in the Error Status register (CHn_ES) and can be caused by any of the following:

- A configuration error, which is an illegal setting in the transfer control descriptor
- An active channel canceled via a "cancel transfer with error" hardware or software request
- A TCD memory error
- An error termination to a bus master read or write cycle

A configuration error is reported when an inconsistent state is represented by one of these factors:

- Starting source or destination address
- Source or destination offsets
- Minor loop byte count
- Transfer size

Each of these possible causes is detailed below:

- The addresses and offsets must be aligned on zero-modulo-transfer-sized boundaries.
- The minor loop byte count must be a multiple of the source and destination transfer sizes.
- All source reads and destination writes must be configured to the natural boundary of the programmed transfer size.

NOTE

To aid in debugging, set the Halt After Error field in the DMA's Control Status register, CSR[HAE]. Upon any error condition, the DMA is halted after the error is recorded. The DMA remains halted and does not process any channel service requests. After the error is fixed, the DMA may be enabled again by clearing the Halt field, CSR[HALT].

- If a scatter/gather operation is enabled upon channel completion, a configuration error is reported if the scatter/gather address (TCDn_DLAST_SGA) is not aligned on a 32-byte boundary.
- If minor loop channel linking is enabled upon channel completion, a configuration error is reported when the link is attempted if the TCDn_CITER[ELINK] field does not equal the TCDn_BITER[ELINK] field.

If enabled, all configuration error conditions, except the scatter/gather and minor-loop link errors, are reported as the channel activates and asserts an error interrupt request. A scatter/gather configuration error is reported when the scatter/gather operation begins at major loop completion if properly enabled. A minor loop channel link configuration error is reported when the link operation is serviced at minor loop completion.

If a system bus read or write is terminated with an error, the data transfer is stopped and the appropriate bus error flag set. In this case, the state of the channel's transfer control descriptor is updated by the eDMA engine with the current source address, destination address, and current iteration count at the point of the fault. When a system bus error occurs, the channel terminates after the next transfer. Due to pipeline effect, the next transfer is already in progress when the bus error is received by the eDMA. If a bus error occurs on the last read prior to beginning the write sequence, the write executes using the data captured during the bus error. If a bus error occurs on the last write prior to switching to the next read sequence, the read sequence executes before the channel terminates due to the destination bus error.

The occurrence of any error causes the eDMA engine to stop normal processing of the active channel immediately (it goes to its error processing states and the transaction to the system bus still has pipeline effect), and the appropriate channel field in the eDMA error register is set to 1. At the same time, the details of the error condition are loaded into the Error Status register (CHn_ES). The major loop complete indicators, setting the transfer control descriptor DONE flag, and the possible assertion of an interrupt request are not affected when an error is detected.

After the error status has been updated, the eDMA engine continues operating by servicing the next appropriate channel. A channel that experiences an error condition is not automatically disabled. If a channel is terminated by an error and then issues another service request before the error is fixed, that channel executes and terminates with the same error condition.

The error status fields are read-only. These error indicators are sticky and cannot be cleared. They show the last recorded error until the DMA is reset. CHn_ES[ERR] is used to determine if a new error condition exists. This field is the logical OR of each channel's error interrupt field (ERR).

After the software has resolved all errors and cleared all of the error interrupt fields, the ES[VLD] is cleared to 0 but the cause of the last error is still indicated.

14.3.4 Channel preemption

The eDMA uses a priority vector value to determine the highest priority channel requesting service.

The priority vector is a combination of:

1. the channel's group priority, CHn_GRPRI
2. the channel's priority level, CHn_PRI[APL]

It can be considered a number composed of these concatenated priority levels: CHn_GRPRI : CHn_PRI[APL]

Priority vector = ((CHn_GRPRI << 8) + (CHn_PRI[APL] << 5) + CHn_*)

A channel requesting service with the highest priority vector value will receive the next execution slot.

An execution slot is available:

1. immediately if the eDMA is idle
2. when an active channel retires
3. when valid preemption conditions exist

NOTE

Preemption is strictly priority based. Preemption is not bound by a specific group number as defined by CHn_GRPRI.

Channel preemption is enabled on a per-channel basis by setting the CHn_PRI[ECP] field. Channel preemption allows the executing channel's data transfers to temporarily suspend in favor of starting a higher-priority channel. After the preempting channel has completed all of its minor loop data transfers, the preempted channel is restored and resumes execution.

After the restored channel completes one read/write sequence, it is again eligible for preemption. If any higher priority channel is requesting service, the restored channel is suspended, and the higher-priority channel is serviced. Nested preemption, that is, attempting to preempt a preempting channel, is not supported. After a preempting channel begins execution, it cannot be preempted.

A channel's ability to preempt another channel can be disabled by setting CHn_PRI[DPA] to 1. When a channel's preempt ability is disabled, that channel cannot suspend a lower-priority channel's data transfer, regardless of the lower-priority channel's ECP setting. This allows for a pool of low-priority, large-data-moving channels to be defined.

You can configure these low-priority channels to not preempt each other, thus preventing a low-priority channel from consuming the preempt slot normally available to a true high-priority channel. When you enable round-robin channel arbitration mode (CSR[ERCA] is set to 1), any channel with a priority level equal to 0 (CHn_PRI[APL] = 0) has preemption disabled and cannot preempt another channel.

14.3.5 Clocking

This module has no clocking considerations.

14.3.6 Interrupts

Software can enable the interrupt for each channel for the following events:

1. The major loop is half complete (INTHALF)
2. The major loop is complete (INTMAJOR)
3. A configuration error occurs (EEI)

14.4 External signals

This module has no external signals.

14.5 Initialization

The following sections discuss initialization of the eDMA and programming considerations.

14.5.1 eDMA initialization

To initialize the eDMA:

1. Write to the CSR if a configuration other than the default is wanted.
2. Write the channel priority levels to the CHn_PRI registers and group priority levels to the CHn_GRPRI registers if a configuration other than the default is wanted.
3. Enable error interrupts in the CHn_CSR[EEI] registers if they are wanted.
4. Write the 32-byte TCD for each channel that may request service.
5. Enable any hardware service requests via the CHn_CSR[ERQ] registers.
6. Request channel service via either:
   - Software: setting TCDn_CSR[START]
   - Hardware: slave device asserting its eDMA peripheral request signal

After any channel requests service, a channel is selected for execution based on the arbitration and priority levels written into the programmer's model. The eDMA engine reads the entire TCD, including the TCD control and status fields, as shown in Table 46, for the selected channel into its internal address path module.

As the TCD is read, the first transfer is initiated on the internal bus, unless a configuration error is detected. Transfers from the source, defined by TCDn_SADDR, to the destination, defined by TCD_DADDR, continue until the number of bytes specified by TCDn_NBYTES are transferred.

When the transfer is complete, the eDMA engine's local TCDn_SADDR, TCDn_DADDR, and TCDn_CITER are written back to the main TCD memory and any minor loop channel linking is performed, if enabled. If the major loop is exhausted, then eDMA executes further post-processing, such as interrupts, major loop channel linking, and scatter/gather operations, if enabled.

Table 46. TCD control and status (TCDn_CSR) fields

TCDn_CSR field name	Description
START	Control field to start the channel explicitly when using a software-initiated DMA service (automatically cleared by hardware)
EEOP	Control field to enable end-of-packet processing
ESDA	Control field to enable storing of the destination address to system memory after the major loop completes
DREQ	Control field to disable hardware-initiated DMA service requests after major loop completion
BWC	Control field for throttling the bandwidth control of a channel
ESG	Control field to enable the scatter-gather feature
INTHALF	Control field to enable interrupt when major loop is half-complete
INTMAJOR	Control field to enable interrupt when major loop completes

Table 47. Channel control and status (CHn_CSR) fields

CHn_CSR field name	Description
ACTIVE	Status field indicating the channel is currently in execution
DONE	Status field indicating major loop completion (cleared by software when a channel begins execution)
EEI	Control field to enable error interrupts
EARQ	Control field to enable external, asynchronous wakeup event in conjunction with the ERQ field
ERQ	Control field to enable hardware service requests

The following figure shows how each DMA request initiates one minor-loop transfer, or iteration, without CPU intervention. DMA arbitration can occur after each minor loop, and one level of minor loop DMA preemption is allowed. The number of minor loops in a major loop is specified by the beginning iteration count (BITER).

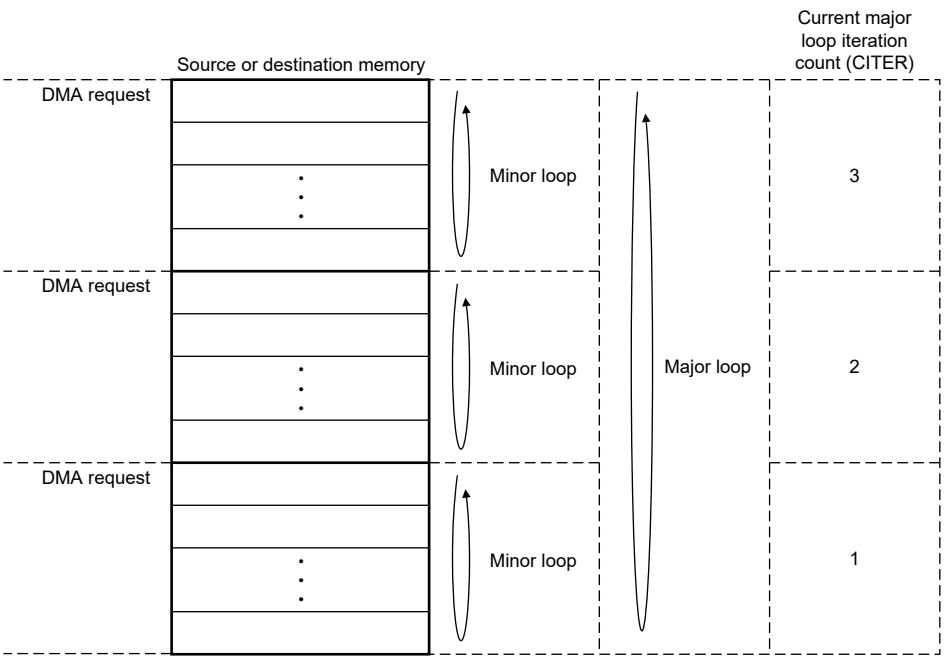


Figure 34. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings are related.

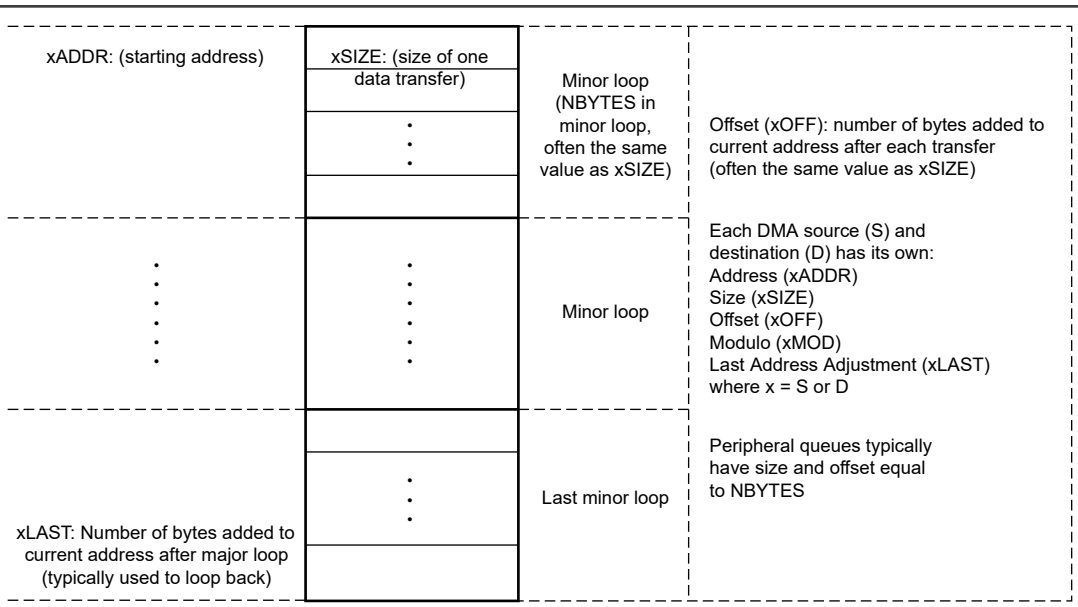


Figure 35. Memory array terms

14.5.2 eDMA arbitration

The eDMA uses a layered arbitration scheme composed of multiple priority levels. The eDMA uses a fixed-priority arbitration scheme with optional round-robin arbitration under specific conditions. The priorities are evaluated in the following order:

Table 48. eDMA arbitration priorities

Priority	Scheme	Description
1 (Highest)	Arbitration group priority	Each channel is assigned an arbitration group via the CHn_GRPRI registers. Priority is given to the highest value (31 being the highest possible value) down to the lowest value (zero, the default).
2	Channel priority	Each channel is assigned a channel priority level via the CHn_PRI registers. The channel priority is a relative priority level within an arbitration group. Priority is given to the highest value (seven being the highest possible value) down to the lowest value (zero, the default). Channel priorities within each arbitration group need not be unique. If multiple channels have the same channel priority level, the channel number will be used to determine priority as defined in row three.
3	Channel number	When two or more channels have the same arbitration group priority and channel priority, the channel number (CHn_NUM) is used to determine the highest priority. Priority is given to the highest channel number. Lowest priority is channel 0. The channel numbers are static and cannot be changed in the programmer's model.
4 (Lowest)	Round-robin	When round-robin is enabled, any channel configured for round-robin operation has lowest priority within an arbitration group. Round-robin is enabled by setting the CSR[ERCA] field to 1. After being enabled, channels with a channel priority of zero (CHn_PRI=0) will use round-robin arbitration. Round-robin arbitration will rotate the channel selection among the channels requesting service with CHn_PRI=0 within the arbitration group. Any non-zero channel within the arbitration group will continue to use fixed-priority arbitration, and if requesting service will be selected over any round-robin channels.

For fixed arbitration, the overall priority can be considered a number composed of three concatenated priority levels: CHn_GRPRI:CHn_PRI:CH_NUM. The largest number has the highest priority and the lowest number has the lowest priority.

For round-robin arbitration, the priority number is CHn_GRPRI:0:X. The module rotates through the CHn_PRI=0 channels requesting service without regard to priority among these channels. Any channel within the arbitration group for which CHn_PRI is greater than 0 will be serviced before the round-robin channels.

14.5.3 Programming errors

The eDMA performs various tests on the transfer control descriptor to verify consistency in the descriptor data.

The channel number causing the error is recorded in the Error Status register (CHn_ES). If the error source is not removed before the next activation of the problematic channel, the error is detected and recorded again. Setting the halt after error field, CSR[HAE], will halt the DMA and prevent recurrence of the error.

14.5.4 Arbitration mode considerations

This section discusses arbitration considerations for eDMA.

14.5.4.1 Fixed group arbitration, fixed channel arbitration

In this mode, eDMA selects for execution the channel service request from the highest-priority channel in the highest-priority group. If eDMA is programmed so that the channels within a high-priority group have a high number of requests or large data transfers, that group may consume all the bandwidth of the eDMA controller. That is, no lower-priority groups are serviced if there is always at least one DMA request pending on a channel in the highest-priority group when the controller arbitrates the next DMA request. The advantage of this scenario is that latency can be small for channels that need to be serviced quickly.

14.5.4.2 Fixed group arbitration, round-robin channel arbitration

The highest-priority group with a request is serviced. Lower-priority groups are serviced if no pending requests exist in the higher-priority groups.

Within each group, channels are serviced starting with the highest non-zero channel priority. For all channels with a channel priority programmed to 0, selection begins with the highest channel number requesting service and then rotates through to the lowest channel number requesting service. The round-robin channel arbitration can provide a fairness mechanism to lower-priority channels.

This scenario could cause the same bandwidth consumption problem as indicated in Fixed group arbitration, fixed channel arbitration, but all the channels in the highest-priority group will be serviced. Service latency is short on the highest-priority group, but could potentially be very much longer as the group priority decreases.

14.5.5 Performing DMA transfers

This section presents examples on how to perform DMA transfers with the eDMA.

14.5.5.1 Single request

To perform a simple transfer of n bytes of data with one activation, set the major loop to one (TCD*n*_CITER = TCD*n*_BITER = 1). The data transfer begins after the channel service request is acknowledged and the channel is selected to execute. After the transfer is complete, the CH*n*_CSR[DONE] field is set to 1 and an interrupt is generated if properly enabled.

For example, the following TCD entry is configured to transfer 16 bytes of data. The eDMA is programmed for one iteration of the major loop transferring 16 bytes per iteration. The source memory has a byte-wide memory port located at 0x1000. The destination memory has a 32-bit port located at 0x2000. The address offsets are programmed in increments to match the transfer size: one byte for the source, and four bytes for the destination. The final source and destination addresses are adjusted to return to their beginning values.

```
TCDn_CITER = TCDn_BITER = 1
TCDn_NBYTES = 16
TCDn_SADDR = 0x1000
TCDn_SOFF = 1
TCDn_ATTR[SSIZE] = 0
TCDn_SLAST = -16
TCDn_DADDR = 0x2000
TCDn_DOFF = 4
TCDn_ATTR[DSIZE] = 2
```

```
TCDn_DLAST_SGA= -16
TCDn_CSR[INTMAJ] = 1
TCDn_CSR[START] = 1 (should be written last after all other fields have been initialized)
All other TCDn fields = 0
```

This generates the following event sequence:

1. User write to the TCD*n*_CSR[START] field requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes:
   - CH*n*_CSR[DONE] = 0
   - TCD*n*_CSR[START] = 0
   - CH*n*_CSR[ACTIVE] = 1
4. eDMA engine reads: channel TCD data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:
   a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
   b. Write 32 bits to location 0x2000 → first iteration of the minor loop.
   c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
   d. Write 32 bits to location 0x2004 → second iteration of the minor loop.
   e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
   f. Write 32 bits to location 0x2008 → third iteration of the minor loop.
   g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
   h. Write 32 bits to location 0x200C → last iteration of the minor loop → major loop complete.
6. The eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 1 (TCD*n*_BITER).
7. The eDMA engine writes: CH*n*_CSR[ACTIVE] = 0, CH*n*_CSR[DONE] = 1, CH*n*_INT[INT] = 1.
8. The channel retires and the eDMA goes idle or services the next channel.

14.5.5.2 Multiple requests

The following example transfers 32 bytes via two hardware requests, but is otherwise the same as the previous example. The only fields that change are the major loop iteration count and the final address offsets. The eDMA is programmed for two iterations of the major loop, transferring 16 bytes per iteration. After the channel's hardware requests are enabled via the CHn_CSR[ERQ] register field, the slave device initiates channel service requests.

```
TCDn_CITER = TCDn_BITER = 2
TCDn_SLAST = -32
TCDn_DLAST_SGA = -32
```

This would generate the following sequence of events:

1. First hardware (eDMA peripheral) requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD*n* data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:

a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
b. Write 32 bits to location 0x2000 → first iteration of the minor loop.
c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
d. Write 32 bits to location 0x2004 → second iteration of the minor loop.
e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
f. Write 32 bits to location 0x2008 → third iteration of the minor loop.
g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
h. Write 32 bits to location 0x200C → last iteration of the minor loop.

6. eDMA engine writes: TCD*n*_SADDR = 0x1010, TCD*n*_DADDR = 0x2010, TCD*n*_CITER = 1.
7. eDMA engine writes: CH*n*_CSR[ACTIVE] = 0.
8. The channel retires, which concludes one iteration of the major loop. The eDMA goes idle or services the next channel.
9. Second hardware (eDMA peripheral) requests channel service.
10. The channel is selected by arbitration for servicing.
11. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
12. eDMA engine reads: Channel TCD data from local memory to internal register file.
13. The source-to-destination transfers are executed as follows:
    a. Read byte from location 0x1010, read byte from location 0x1011, read byte from 0x1012, read byte from 0x1013.
    b. Write 32 bits to location 0x2010 → first iteration of the minor loop.
    c. Read byte from location 0x1014, read byte from location 0x1015, read byte from 0x1016, read byte from 0x1017.
    d. Write 32 bits to location 0x2014 → second iteration of the minor loop.
    e. Read byte from location 0x1018, read byte from location 0x1019, read byte from 0x101A, read byte from 0x101B.
    f. Write 32 bits to location 0x2018 → third iteration of the minor loop.
    g. Read byte from location 0x101C, read byte from location 0x101D, read byte from 0x101E, read byte from 0x101F.
    h. Write 32 bits to location 0x201C → last iteration of the minor loop → major loop complete.
14. eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 2 (TCD*n*_BITER).
15. eDMA engine writes: CH*n*_CSR[ACTIVE] = 0, CH*n*_CSR[DONE] = 1, CH*n*_INT[INT] = 1.
16. The channel retires, which concludes with the major loop complete. The eDMA goes idle or services the next channel.

14.5.5.3 Using the modulo feature

The modulo feature of the eDMA allows implementation of a circular data queue in which the size of the queue is a power of 2. xMOD is a 5-bit field for the source and destination in the TCD, and it specifies which lower address bits increment from their original value after the address+offset calculation. All upper address bits remain the same as in the original value. A setting of 0 for this field disables the modulo feature. Modulo addressing applies to cases where the minor loop offset is enabled; that is, the upper address bits remain the same after the minor loop offset is added to the source or destination address.

The following table shows how the transfer addresses are specified based on the setting of the MOD field. Here a circular buffer is created where the address wraps to the original value but the 28 upper address bits (0x1234567*x*) retain their original value. In this example, the source address is set to 0x12345670, the offset is set to four bytes, and the MOD field is set to four, which allows for a 2^4 byte (16 byte) queue size.

Table 49. Modulo example

Transfer number	Address
1	0x12345670
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

14.5.6 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

14.5.6.1 Testing for minor loop completion

There are two methods to test for minor loop completion when using software-initiated service requests.

1. The first method is to read the TCDn_CITER field and test for a change.
2. The second method, extracted from the sequence shown below, is to test the TCDn_CSR[START] field and the CHn_CSR[ACTIVE] field. The minor-loop-complete condition is indicated by both fields reading 0 after TCDn_CSR[START] is set to 1. Polling the CHn_CSR[ACTIVE] field only may be inconclusive because the active status may be missed if the channel execution is short in duration.

The CHn_CSR and TCDn_CSR status fields execute the following sequence for a software-activated channel:

Stage	TCDn_CSR field	CHn_CSR fields		State
	START	ACTIVE	DONE	
1	1	0	0	Initiate channel service request via software.
2	0	1	0	Channel is executing.
3a	0	0	0	Channel has completed the minor loop and is idle.
3b	0	0	1	Channel has completed the major loop and is idle.

The best method to test for minor-loop completion when using hardware-initiated (that is, peripheral-initiated) service requests is to read the TCDn_CITER field and test for a change. The hardware request and acknowledge handshake signals are not visible in the programmer's model.

The TCD status fields execute the following sequence for a hardware-activated channel:

Stage	TCDn_CSR field	CHn_CSR fields		State
	START	ACTIVE	DONE	
1	0	0	0	Initiate channel service request via hardware (peripheral request asserted).
2	0	1	0	Channel is executing.
3a	0	0	0	Channel has completed the minor loop and is idle.
3b	0	0	1	Channel has completed the major loop and is idle.

For both activation types, the major-loop-complete status is explicitly indicated via the CHn_CSR[DONE] field.

The TCDn_CSR[START] field is cleared to 0 automatically when the channel begins execution, regardless of how the channel activates.

14.5.6.2 Reading the transfer descriptors of active channels

The eDMA reads back the true TCDn_SADDR, TCDn_DADDR, and TCDn_NBYTES values if they are read when a channel executes. The true values of SADDR, DADDR, and NBYTES are the values the eDMA engine currently uses in its internal register file, and not the values in the TCD local memory for that channel. The addresses, SADDR and DADDR, and NBYTES (which decrements to zero as the transfer progresses), can give an indication of the progress of the transfer. All other values are read back from the TCD local memory.

14.5.6.3 Checking channel preemption status

A preemptive situation is one in which a preempt-enabled channel is executing and a higher-priority request becomes active. When round-robin channel arbitration mode is enabled, all channels with their channel priority set to 0 lose their preempt ability. Channel priorities of 0 are treated as equal, that is, they are constantly rotating, when round-robin arbitration mode is enabled.

The CHn_CSR[ACTIVE] field for the preempted channel remains asserted throughout the preemption. The preempted channel is temporarily suspended when the preempting channel executes one major loop iteration. If two CHn_CSR[ACTIVE] fields are set simultaneously in the global TCD map, a higher-priority channel is actively preempting a lower-priority channel.

14.5.7 Channel linking

Channel linking (or chaining) is a mechanism in which one channel sets the TCDn_CSR[START] field of another channel (or itself), thus initiating a service request for that channel. When properly enabled, the eDMA engine automatically performs this operation at the major or minor loop completion.

The minor loop channel linking occurs at the completion of the minor loop (or one iteration of the major loop). The TCDn_CITER[ELINK] field determines whether a minor loop link is requested. When enabled, the channel link is made after each iteration of the major loop except for the last. When the major loop is exhausted, only the major loop channel link fields are used to determine if a channel link should be made. For example, using an initial field setting of:

```
MP_CSR[GCLC] = 1
TCDn_CITER[ELINK] = 1
TCDn_CITER[LINKCH] = 0xC
TCDn_CITER[CITER] value = 0x4
TCDn_CSR[MAJORELINK] = 1
TCDn_CSR[MAJORLINKCH] = 0x7
```

executes as:

1. Minor loop done → set TCD12_CSR[START] field
2. Minor loop done → set TCD12_CSR[START] field
3. Minor loop done → set TCD12_CSR[START] field
4. Minor loop done, major loop done→ set TCD7_CSR[START] field

When minor loop linking is enabled (TCDn_CITER[ELINK] = 1), the TCDn_CITER[CITER] field uses a nine-bit vector to form the current iteration count. When minor loop linking is disabled (TCDn_CITER[ELINK] = 0), the TCDn_CITER[CITER] field uses a 15-bit vector to form the current iteration count. The bits associated with the TCDn_CITER[LINKCH] field are concatenated onto the CITER value to increase the range of the CITER.

NOTE

The TCDn_CITER[ELINK] field and the TCDn_BITER[ELINK] field must be equal — if they are not, a configuration error is reported. The CITER and BITER vector widths must be equal to calculate the major loop halfway done interrupt point.

The following table summarizes how a DMA channel can link to another DMA channel, that is, use another channel's TCD, at the end of a loop.

Table 50. Channel linking parameters

Wanted link behavior	TCD control field name	Description
Link at end of minor loop	TCDn_CITER[ELINK]	Enable channel-to-channel linking on minor loop completion (current iteration)
	TCDn_CITER[LINKCH]	Link channel number when linking at end of minor loop (current iteration)
Link at end of major loop	TCDn_CSR[MAJORELINK]	Enable channel-to-channel linking on major loop completion
	TCDn_CSR[MAJORLINKCH]	Link channel number when linking at end of major loop

14.5.8 Dynamic programming

This section provides recommended methods to change the programming model during channel execution.

Steps for channel linking:

1. Set GCLC= 1
2. For linking after major loop complete:
   a. Set MAJORLINKCH = target channel number
   b. Set MAJORELINK = 1, enable the major loop link
   c. Upon major loop completion (CITER decrements to 0), the link will be made.
3. For linking on minor loop completion:
   a. Set LINKCH = target channel number
   b. Set ELINK = 1, enable the minor loop link
   c. Set the beginning iteration count to the same values; LINKCH = target channel and ELINK = 1,
   d. Upon minor loop completion (that is, after each service request except when the major loop is complete) the link will be made.

14.5.8.1 Dynamically changing the channel priority

To change group or channel priority levels:

1. Halt the DMA by writing 1 to the CSR[HALT] field.
2. Change the group or channel priorities as wanted.
3. Enable normal DMA operations by writing 0 to the CSR[HALT] field.

14.5.8.2 Dynamic channel linking

Dynamic channel linking is the process of setting the TCDn_CSR[MAJORELINK] field during channel execution (see the diagram in TCD structure). This field is read from the TCD local memory at the end of channel execution, thus allowing you to enable the feature during channel execution.

Because you are allowed to change the configuration during execution, you need a coherency model. Consider the scenario where you attempt to execute a dynamic channel link by enabling the TCDn_CSR[MAJORELINK] field at the same time the eDMA engine is retiring the channel. TCDn_CSR[MAJORELINK] would be set in the programmer's model, but it would be unclear whether the actual link was made before the channel retired.

We recommend that you use the following coherency model when executing a dynamic channel link request.

1. Write 1 to the TCDn_CSR[MAJORELINK] field.
2. Read back the TCDn_CSR[MAJORELINK] field.
3. Test the TCDn_CSR[MAJORELINK] request status:
   - If TCDn_CSR[MAJORELINK] = 1, the dynamic link attempt was successful.
   - If TCDn_CSR[MAJORELINK] = 0, the attempted dynamic link did not succeed (the channel was already retiring).

For this request, the TCD local memory controller forces the TCDn_CSR[MAJORELINK] field to 0 on any writes to a channel's TCDn_CSR[7:0] after that channel's CHn_CSR[DONE] field is set to 1, indicating the major loop is complete.

NOTE

You must clear the CHn_CSR[DONE] field to 0 before writing to the TCDn_CSR[MAJORELINK] field. The CHn_CSR[DONE] field is cleared to 0 automatically by the eDMA engine after a channel begins execution.

14.5.8.3 Dynamic scatter/gather

Scatter/gather is the process of automatically loading a new TCD into a channel. It allows a DMA channel to use multiple TCDs; this enables a DMA channel to scatter the DMA data to multiple destinations or gather it from multiple sources. When scatter/gather is enabled and the channel has finished its major loop, a new TCD is fetched from system memory and loaded into that channel's descriptor location in the eDMA programmer's model, thus replacing the current descriptor.

Because you are allowed to change the configuration during execution, you need a coherency model. Consider the scenario where you attempt to execute a dynamic scatter/gather operation by enabling the TCDn_CSR[ESG] field at the same time the eDMA engine is retiring the channel. The TCDn_CSR[ESG] field would be set in the programmer's model, but it would be unclear whether the actual scatter/gather request was honored before the channel retired.

Two methods are recommended for executing a dynamic scatter/gather request. Whenever the TCDn_CSR is written, the TCD local memory controller forces the TCDn_CSR[ESG] field to 0 on any writes to a channel's TCDn_CSR[7:0] after that channel's CHn_CSR[DONE] field has been set to 1, indicating the major loop is complete. If attempting to set the ESG, ensure the DONE field is cleared to 0.

NOTE

You must clear the CHn_CSR[DONE] field to 0 before writing the TCDn_CSR[MAJORELINK] or TCDn_CSR[ESG] fields. The CHn_CSR[DONE] field is cleared to 0 automatically by the eDMA engine after a channel begins execution and is set to 1 upon major loop completion.

14.5.8.3.1 Method 1 (channel not using major loop channel linking)

For a channel not using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request.

When the TCDn_CSR[MAJORELINK] field is 0, the TCDn_CSR[MAJORLINKCH] field is not used by the eDMA. In this case, the TCDn_CSR[MAJORLINKCH] bits may be used for other purposes. This method uses the TCDn_CSR[MAJORLINKCH] field as a TCDn_CSR identification (ID).

When the descriptors are built, write a unique TCDn_CSR ID in the TCDn_CSR[MAJORLINKCH] field for each TCDn_CSR associated with a channel using dynamic scatter/gather.

1. Write a 1 to the TCDn_CSR[DREQ] field. Should a dynamic scatter/gather attempt fail, setting the TCDn_CSR[DREQ] field to 1 will prevent future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a DLAST final offset value.
2. Write the TCDn_DLAST_SGA field with the scatter/gather address.
3. Write a 1 to the TCDn_CSR[ESG] field.
4. Read back the 16-bit TCDn_CSR control/status field.
5. Test the TCDn_CSR[ESG] request status and TCDn_CSR[MAJORLINKCH] value:
   - If ESG = 1, the dynamic scatter/gather attempt was successful.
   - If ESG = 0 and the MAJORLINKCH (ID) did not change, the dynamic scatter/gather attempt was not successful (the channel was already retiring).
   - If ESG = 0 and the MAJORLINKCH (ID) changed, the dynamic scatter/gather attempt was successful (the new TCDn_CSR's ESG value cleared the ESG field to 0).

14.5.8.3.2 Method 2 (channel using major loop channel linking)

For a channel using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request. This method uses the TCDn_DLAST_SGA field as a TCD identification (ID).

1. Write a 1 to the TCDn_CSR[DREQ] field. Should a dynamic scatter/gather attempt fail, setting the DREQ field to 1 will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (DADDR) that was calculated using a scatter/gather address (written in the next step) instead of a DLAST final offset value.
2. Write the TCDn_DLAST_SGA field with the scatter/gather address.
3. Write a 1 to the TCDn_CSR[ESG] field.
4. Read back the TCDn_CSR[ESG] field.
5. Test the TCDn_CSR[ESG] request status:
   - If ESG = 1, the dynamic scatter/gather attempt was successful.
   - If ESG = 0, read the 32-bit TCDn_DLAST_SGA field.
   - If ESG = 0 and the TCDn_DLAST_SGA did not change, the dynamic scatter/gather attempt was not successful (the channel was already retiring).
   - If ESG = 0 and the TCDn_DLAST_SGA changed, the dynamic scatter/gather attempt was successful (the new TCDn_CSR's ESG value cleared the ESG field to 0).

14.5.9 Suspend/resume a DMA channel with active hardware service requests

The DMA allows you to move data from memory or peripheral registers to another location in memory or to peripheral registers without CPU interaction. After the DMA and peripherals are configured and active, it is rare but supported to suspend a peripheral's service request dynamically. In this scenario, there are certain restrictions to disabling a DMA hardware service request. For

coherency, you must follow a specific procedure. This section provides guidance on how to coherently suspend and resume a Direct Memory Access (DMA) channel when the DMA is triggered by a slave module such as the Serial Peripheral Interface (DSPI), Sigma Delta Analog to Digital Convertor (SDADC), or other module.

14.5.9.1 Suspend an active DMA channel

To suspend an active DMA channel:

1. Stop the DMA service request at the peripheral first. Confirm it has been disabled by reading back the appropriate register in the peripheral.
2. Check the DMA's Hardware Request Status (HRS) to ensure there is no service request to the DMA channel being suspended. Then disable the hardware service request by clearing the ERQ field to 0 on the appropriate DMA channel.

For example, assume the DSPI is set as a master for transmitting data via a DMA service request when the TXFIFO has an empty slot. The DMA will transfer the next command and data to the TXFIFO upon the request. If you need to suspend the DMA/DSPI transfer loop, perform the following steps:

1. Disable the DMA service request at the source by writing 0 to DSPI_RSER[TFFF_RE]. Confirm that DSPI_RSER[TFFF_RE] is 0.
2. Ensure there is no DMA service request from the DSPI by verifying that HRS[HRS] is 0 for the appropriate channel. If no service request is present, disable the DMA channel by clearing the channel's ERQ field to 0. If a service request is present, wait until the request has been processed and the HRS field reads 0.

14.5.9.2 Resume a DMA channel

To resume a DMA channel:

1. Enable the DMA service request on the appropriate channel by setting its ERQ field to 1.
2. Enable the DMA service request at the peripheral.

14.6 Memory map/register definition

The eDMA programming model is partitioned into three parts:

1. The first part defines a number of registers providing overall control functions and is known as the management page.
2. The second part corresponds to the channel (CH) control, status, and configuration.
3. The third part corresponds to the local TCD memory.

TCD memory

Each channel requires a 32-byte transfer control descriptor for defining the data movement operation. Each TCDn definition is presented as 11 registers of 16 or 32 bits. See TCD register descriptions for details.

TCD initialization

The TCD memory is in an unknown state after reset. Only the TCD START bit is initialized to 0. Prior to activating a channel, you must initialize its TCD with the appropriate transfer profile.

TCD structure

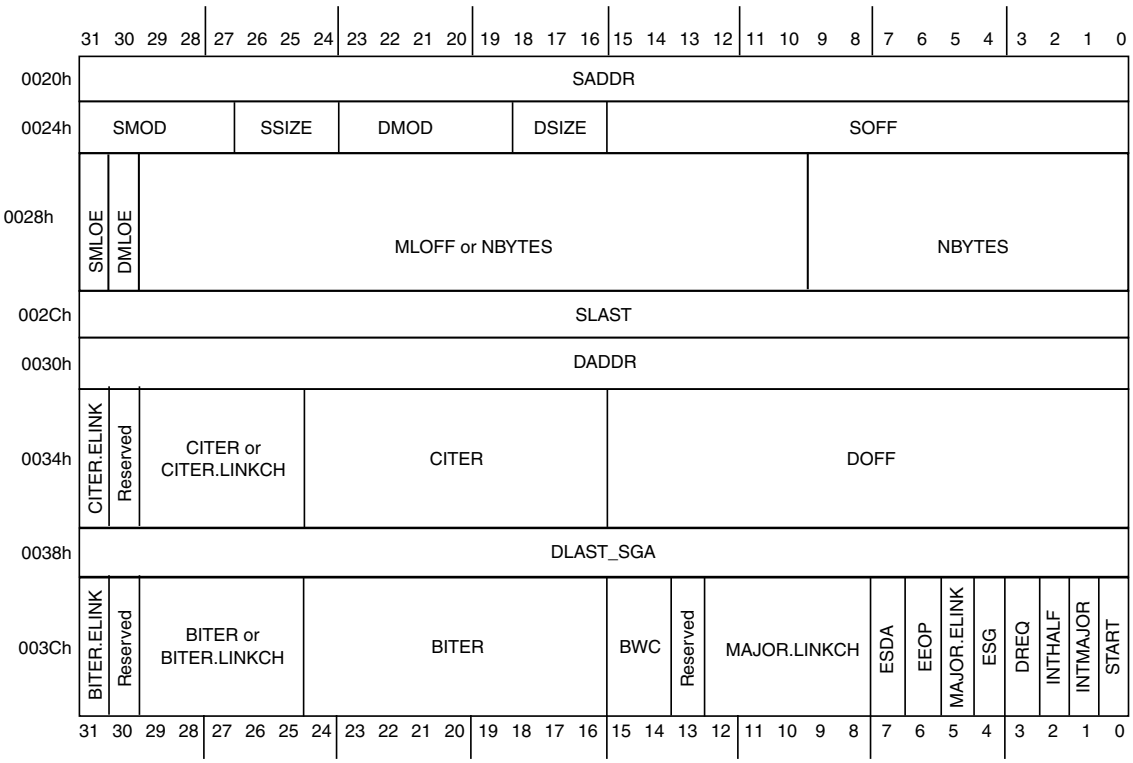


Figure 36. TCD structure

Accesses to reserved memory and fields

- Reading reserved fields in a register returns the value of zero.
- Writes to reserved fields in a register are ignored.
- Reading or writing a reserved memory location generates a bus error.

14.6.1 eDMA register descriptions

14.6.1.1 eDMA memory map

eDMA base address: 4020_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Management Page Control (CSR)	32	RW	0030_0000h
4h	Management Page Error Status (ES)	32	R	0000_0000h
8h	Management Page Interrupt Request Status (INT)	32	R	0000_0000h
Ch	Management Page Hardware Request Status (HRS)	32	R	0000_0000h
100h - 17Ch	Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI)	32	RW	0000_0000h

14.6.1.2 Management Page Control (CSR)

Offset

Register	Offset
CSR	0h

Function

The Management Page Control register defines the basic operating configuration of the DMA.

Arbitration uses a two-tier priority system; group and channel priority. The eDMA assigns each channel to a priority group. Group arbitration is fixed-priority and cannot be changed. Channel arbitration uses fixed priority and may be configured to use a selective round-robin scheme for specified channels within each priority group. For fixed-priority arbitration, eDMA selects for execution the highest priority channel requesting service in the highest priority arbitration group.

The channel priority registers assign the relative priorities within each arbitration group; see CHn_PRI. All channels with a non-zero CHn_PRI value use fixed-priority arbitration.

When you enable round-robin arbitration, all channels with channel priority set to zero do not have a priority and, of those channels requesting service, are cycled through (from high to low channel number) without regard to priority relative to each other within the same priority group. Any channel with a non-zero CHn_PRI value automatically has a higher priority over the round-robin channels. A channel's priority group is assigned in Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI).

NOTE

For correct operation, changes to the CSR[ERCA, GCLC, GMRC] fields must be performed when the DMA channels are inactive; that is, when the CSR[ACTIVE] field is 0.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						CX	ECX	GMRC	GCLC	HALT	HAE	Reserved	ERCA	EDBG	Reserved
W	0												0			0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ACTIVE	DMA Active Status 0b - eDMA is idle 1b - eDMA is executing a channel

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-29 —	Reserved
28-24 ACTIVE_ID	Active Channel ID This field identifies the channel number that is executing when the ACTIVE bit is 1.
23-16 —	Reserved
15-10 —	Reserved
9 CX	Cancel Transfer When set to 1, this field cancels the remaining data transfer, stops the executing channel, and forces the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. CX clears itself to 0 after the cancel has been honored. This cancel retires the channel normally as if the minor loop had been completed. 0b - Normal operation 1b - Cancel the remaining data transfer
8 ECX	Cancel Transfer With Error Cancellation of the remaining data transfer is similar to that of the CX field. Execution of the channel is stopped and the minor loop is forced to finish. The cancellation takes effect after the last write of the current read/write sequence. The ECX field clears itself to 0 after the cancel is honored. In addition to cancelling the transfer, ECX treats the cancel as an error condition, thus updating Management Page Error Status (ES) and generating an optional error interrupt. 0b - Normal operation 1b - Cancel the remaining data transfer
7 GMRC	Global Master ID Replication Control **NOTE** If master ID replication is disabled, the privileged protection level (Supervisor mode) for DMA transfers is used. 0b - Master ID replication disabled for all channels 1b - Master ID replication available and controlled by each channel's CHn_SBR[EMI] setting
6 GCLC	Global Channel Linking Control 0b - Channel linking disabled for all channels 1b - Channel linking available and controlled by each channel's link settings
5	Halt DMA Operations

Table continues on the next page...

Table continued from the previous page...

Field	Function
HALT	This field stalls the start of any new channels. Executing channels are allowed to complete. Channel execution resumes when this field is cleared to 0. 0b - Normal operation 1b - Stall the start of any new channels
4 HAE	Halt After Error When this field is set to 1, any error causes the HALT field to be set to 1. Then all service requests are ignored until the HALT field is cleared to 0. 0b - Normal operation 1b - Any error causes the HALT field to be set to 1
3 —	Reserved
2 ERCA	Enable Round Robin Channel Arbitration 0b - Round-robin channel arbitration disabled. Fixed priority arbitration used for channel selection within each group 1b - Round-robin channel arbitration enabled. Round-robin arbitration used for channel selection within each group
1 EDBG	Enable Debug When in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete. DMA resumes channel execution when the system exits debug mode or clears the EDBG field to 0. 0b - Debug mode disabled. When in debug mode, the DMA continues to operate 1b - Debug mode is enabled. When in debug mode, the DMA stalls the start of a new channel
0 —	Reserved

14.6.1.3 Management Page Error Status (ES)

Offset

Register	Offset
ES	4h

Function

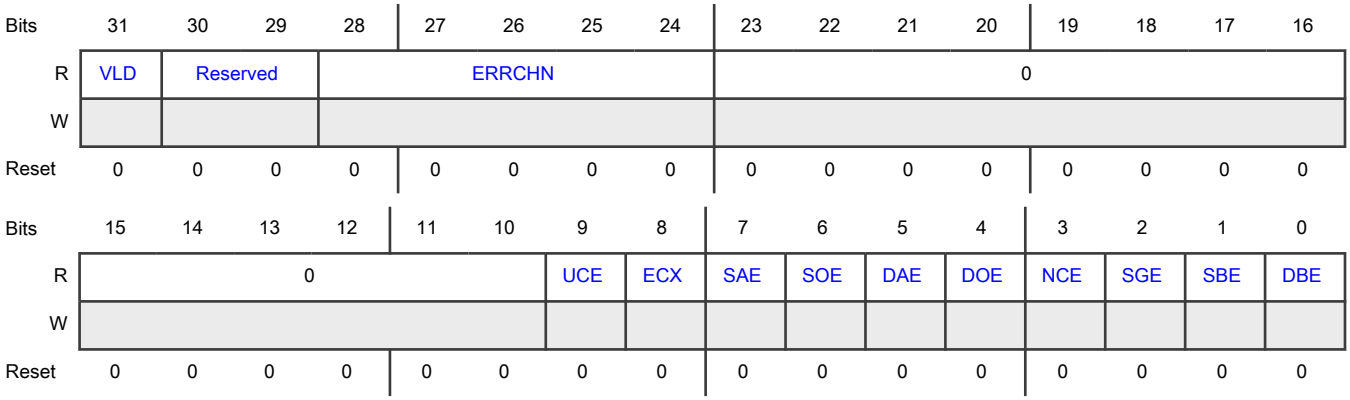
The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- An illegal setting in the transfer control descriptor
- An error termination to a bus master read or write cycle

- An uncorrectable error that occurred when the device was accessing the TCD SRAM
- A "cancel transfer with error" request was made via the corresponding cancel transfer field or input signal

Upon any error condition, the software must initialize the TCD of the channel that contains the error, as it is in an incomplete state after an error. See Fault reporting and handling for more details.

Diagram



Fields

Field	Function
31 VLD	Valid Logical OR of all CHn_ES[ERR] status fields. 0b - No CHn_ES[ERR] fields are set to 1 1b - At least one CHn_ES[ERR] field is set to 1, indicating a valid error exists that software has not cleared
30-29 —	Reserved
28-24 ERRCHN	Error Channel Number or Canceled Channel Number The channel number of the last recorded error or last recorded error-canceled transfer.
23-10 —	Reserved
9 UCE	Uncorrectable TCD Error During Channel Execution UCE is set to 1 only when an uncorrectable ECC error occurs on an access generated by the DMA. If a CPU access to the TCD causes an uncorrectable ECC error, then that access receives a bus error response. **NOTE** When the eDMA sees a RAM error on an IPS access (when you are accessing a TCD), it reports the error as a bus abort. When the DMA engine receives a RAM error (the execution engine is accessing a TCD) it is recorded in the Error Status register, ES[UCE], along with the channel number.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No uncorrectable ECC error 1b - Last recorded error was an uncorrectable TCD RAM error
8 ECX	Transfer Canceled The ECX operation is a management page function. When employed, the targeted channel's CHn_ES register reports an unspecified error; that is, only the CHn_ES[ERR] field is set to 1. The management page has full view of the error condition. 0b - No canceled transfers 1b - Last recorded entry was a canceled transfer by the error cancel transfer input
7 SAE	Source Address Error When this field is 1, it indicates that TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SADDR field
6 SOE	Source Offset Error When this field is 1, it indicates that TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SOFF field
5 DAE	Destination Address Error When this field is 1, it indicates that TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DADDR field
4 DOE	Destination Offset Error When this field is 1, it indicates that TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DOFF field
3 NCE	NBYTES/CITER Configuration Error This error indicates that one of the following has occurred: • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE] • TCDn_CITER[CITER] is equal to zero • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK] 0b - No NBYTES/CITER configuration error 1b - The last recorded error was NBYTES equal to zero or a CITER not equal to BITER error. Last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 SGE	Scatter/Gather Configuration Error When this field is 1, it indicates that TCDn_DLAST_SGA is not on a 32-byte boundary. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG] is enabled. 0b - No scatter/gather configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DLAST_SGA field
1 SBE	Source Bus Error 0b - No source bus error 1b - Last recorded error was a bus error on a source read
0 DBE	Destination Bus Error 0b - No destination bus error 1b - Last recorded error was a bus error on a destination write

14.6.1.4 Management Page Interrupt Request Status (INT)

Offset

Register	Offset
INT	8h

Function

This register shows the current state of the interrupt service requests for all eDMA channels.

Diagram

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

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

Fields

Field	Function
31-0 INT	Interrupt Request Status The INT register presents the interrupt request status for each eDMA channel. Depending on the appropriate field setting in the transfer control descriptors, the eDMA engine generates an interrupt on data transfer completion or an error condition. The eDMA routes channel interrupt requests to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate field in the channel's interrupt request register, CHn_INT, thus negating the interrupt request. 0b - Interrupt request for corresponding channel not present 1b - Interrupt request for corresponding channel present

14.6.1.5 Management Page Hardware Request Status (HRS)

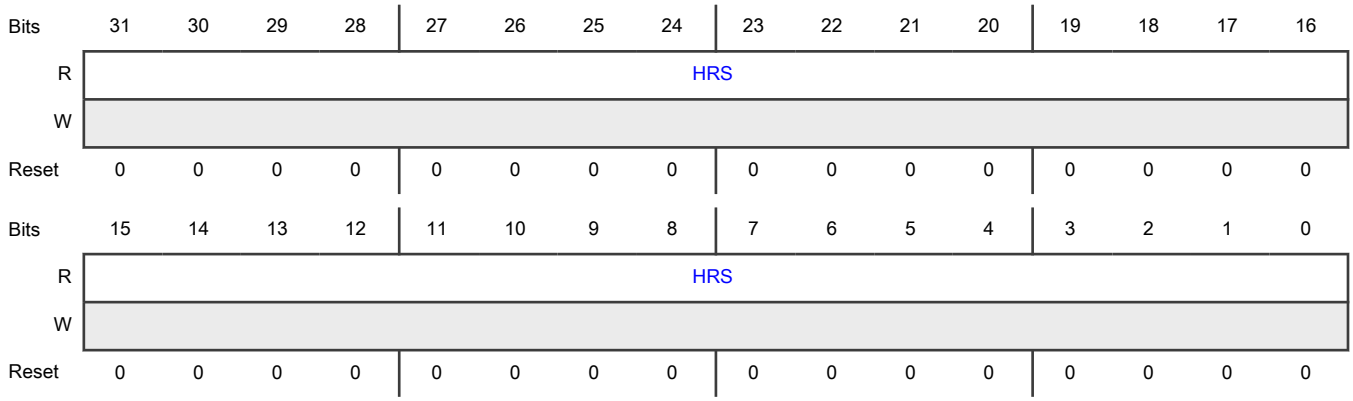
Offset

Register	Offset
HRS	Ch

Function

The hardware request status register (HRS) shows the current state of the hardware service request signaling as seen by eDMA's arbitration logic.

Diagram



Fields

Field	Function
31-0 HRS	Hardware Request Status The HRS bit for its respective channel remains asserted for the period when a hardware request is present on the channel.

Table continues on the next page...

Field	Function
	0b - Hardware service request for corresponding channel is not present 1b - Hardware service request for corresponding channel is present

14.6.1.6 Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI)

Offset

For n = 0 to 31:

Register	Offset
CHn_GRPRI	100h + (n × 4h)

Function

The contents of this register define the arbitration group associated with each channel. Using a fixed-priority group arbitration scheme, eDMA evaluates the arbitration group priorities by numeric value from highest group number to lowest; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, and so on. The range of the group priority values is limited to the values of 0 through 31. Within each arbitration group, the channel priority assignment CH*n*_PRI determines the highest-priority channel.

Diagram

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

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

Fields

Field	Function
31-5 —	Reserved
4-0 GRPRI	Arbitration Group For Channel n Fixed-priority arbitration group number.

14.6.2 TCD register descriptions

14.6.2.1 TCD memory map

TCD base address: 4021_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Channel Control and Status (CH0_CSR)	32	RW	0000_0000h
4h	Channel Error Status (CH0_ES)	32	RW	0000_0000h
8h	Channel Interrupt Status (CH0_INT)	32	RW	0000_0000h
Ch	Channel System Bus (CH0_SBR)	32	RW	0000_8002h
10h	Channel Priority (CH0_PRI)	32	RW	0000_0000h
20h	TCD Source Address (TCD0_SADDR)	32	RW	0000_0000h
24h	TCD Signed Source Address Offset (TCD0_SOFF)	16	RW	0000h
26h	TCD Transfer Attributes (TCD0_ATTR)	16	RW	0000h
28h	TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO)	32	RW	0000_0000h
28h	TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES)	32	RW	0000_0000h
2Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA)	32	RW	0000_0000h
30h	TCD Destination Address (TCD0_DADDR)	32	RW	0000_0000h
34h	TCD Signed Destination Address Offset (TCD0_DOFF)	16	RW	0000h
36h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO)	16	RW	0000h
36h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES)	16	RW	0000h
38h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA)	32	RW	0000_0000h
3Ch	TCD Control and Status (TCD0_CSR)	16	RW	0000h
3Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO)	16	RW	0000h
3Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES)	16	RW	0000h
4000h	Channel Control and Status (CH1_CSR)	32	RW	0000_0000h
4004h	Channel Error Status (CH1_ES)	32	RW	0000_0000h
4008h	Channel Interrupt Status (CH1_INT)	32	RW	0000_0000h
400Ch	Channel System Bus (CH1_SBR)	32	RW	0000_8002h
4010h	Channel Priority (CH1_PRI)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4020h	TCD Source Address (TCD1_SADDR)	32	RW	0000_0000h
4024h	TCD Signed Source Address Offset (TCD1_SOFF)	16	RW	0000h
4026h	TCD Transfer Attributes (TCD1_ATTR)	16	RW	0000h
4028h	TCD Transfer Size Without Minor Loop Offsets (TCD1_NBYTES_MLOFFNO)	32	RW	0000_0000h
4028h	TCD Transfer Size with Minor Loop Offsets (TCD1_NBYTES_MLOFFYES)	32	RW	0000_0000h
402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD1_SLAST_SDA)	32	RW	0000_0000h
4030h	TCD Destination Address (TCD1_DADDR)	32	RW	0000_0000h
4034h	TCD Signed Destination Address Offset (TCD1_DOFF)	16	RW	0000h
4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD1_CITER_ELINKNO)	16	RW	0000h
4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD1_CITER_ELINKYES)	16	RW	0000h
4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD1_DLAST_SGA)	32	RW	0000_0000h
403Ch	TCD Control and Status (TCD1_CSR)	16	RW	0000h
403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD1_BITER_ELINKNO)	16	RW	0000h
403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD1_BITER_ELINKYES)	16	RW	0000h
8000h	Channel Control and Status (CH2_CSR)	32	RW	0000_0000h
8004h	Channel Error Status (CH2_ES)	32	RW	0000_0000h
8008h	Channel Interrupt Status (CH2_INT)	32	RW	0000_0000h
800Ch	Channel System Bus (CH2_SBR)	32	RW	0000_8002h
8010h	Channel Priority (CH2_PRI)	32	RW	0000_0000h
8020h	TCD Source Address (TCD2_SADDR)	32	RW	0000_0000h
8024h	TCD Signed Source Address Offset (TCD2_SOFF)	16	RW	0000h
8026h	TCD Transfer Attributes (TCD2_ATTR)	16	RW	0000h
8028h	TCD Transfer Size Without Minor Loop Offsets (TCD2_NBYTES_MLOFFNO)	32	RW	0000_0000h
8028h	TCD Transfer Size with Minor Loop Offsets (TCD2_NBYTES_MLOFFYES)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD2_SLAST_SDA)	32	RW	0000_0000h
8030h	TCD Destination Address (TCD2_DADDR)	32	RW	0000_0000h
8034h	TCD Signed Destination Address Offset (TCD2_DOFF)	16	RW	0000h
8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD2_CITER_ELINKNO)	16	RW	0000h
8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD2_CITER_ELINKYES)	16	RW	0000h
8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD2_DLAST_SGA)	32	RW	0000_0000h
803Ch	TCD Control and Status (TCD2_CSR)	16	RW	0000h
803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD2_BITER_ELINKNO)	16	RW	0000h
803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD2_BITER_ELINKYES)	16	RW	0000h
C000h	Channel Control and Status (CH3_CSR)	32	RW	0000_0000h
C004h	Channel Error Status (CH3_ES)	32	RW	0000_0000h
C008h	Channel Interrupt Status (CH3_INT)	32	RW	0000_0000h
C00Ch	Channel System Bus (CH3_SBR)	32	RW	0000_8002h
C010h	Channel Priority (CH3_PRI)	32	RW	0000_0000h
C020h	TCD Source Address (TCD3_SADDR)	32	RW	0000_0000h
C024h	TCD Signed Source Address Offset (TCD3_SOFF)	16	RW	0000h
C026h	TCD Transfer Attributes (TCD3_ATTR)	16	RW	0000h
C028h	TCD Transfer Size Without Minor Loop Offsets (TCD3_NBYTES_MLOFFNO)	32	RW	0000_0000h
C028h	TCD Transfer Size with Minor Loop Offsets (TCD3_NBYTES_MLOFFYES)	32	RW	0000_0000h
C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD3_SLAST_SDA)	32	RW	0000_0000h
C030h	TCD Destination Address (TCD3_DADDR)	32	RW	0000_0000h
C034h	TCD Signed Destination Address Offset (TCD3_DOFF)	16	RW	0000h
C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD3_CITER_ELINKNO)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD3_CITER_ELINKYES)	16	RW	0000h
C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD3_DLAST_SGA)	32	RW	0000_0000h
C03Ch	TCD Control and Status (TCD3_CSR)	16	RW	0000h
C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD3_BITER_ELINKNO)	16	RW	0000h
C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD3_BITER_ELINKYES)	16	RW	0000h
1_0000h	Channel Control and Status (CH4_CSR)	32	RW	0000_0000h
1_0004h	Channel Error Status (CH4_ES)	32	RW	0000_0000h
1_0008h	Channel Interrupt Status (CH4_INT)	32	RW	0000_0000h
1_000Ch	Channel System Bus (CH4_SBR)	32	RW	0000_8002h
1_0010h	Channel Priority (CH4_PRI)	32	RW	0000_0000h
1_0020h	TCD Source Address (TCD4_SADDR)	32	RW	0000_0000h
1_0024h	TCD Signed Source Address Offset (TCD4_SOFF)	16	RW	0000h
1_0026h	TCD Transfer Attributes (TCD4_ATTR)	16	RW	0000h
1_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD4_NBYTES_MLOFFNO)	32	RW	0000_0000h
1_0028h	TCD Transfer Size with Minor Loop Offsets (TCD4_NBYTES_MLOFFYES)	32	RW	0000_0000h
1_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD4_SLAST_SDA)	32	RW	0000_0000h
1_0030h	TCD Destination Address (TCD4_DADDR)	32	RW	0000_0000h
1_0034h	TCD Signed Destination Address Offset (TCD4_DOFF)	16	RW	0000h
1_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD4_CITER_ELINKNO)	16	RW	0000h
1_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD4_CITER_ELINKYES)	16	RW	0000h
1_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD4_DLAST_SGA)	32	RW	0000_0000h
1_003Ch	TCD Control and Status (TCD4_CSR)	16	RW	0000h
1_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD4_BITER_ELINKNO)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD4_BITER_ELINKYES)	16	RW	0000h
1_4000h	Channel Control and Status (CH5_CSR)	32	RW	0000_0000h
1_4004h	Channel Error Status (CH5_ES)	32	RW	0000_0000h
1_4008h	Channel Interrupt Status (CH5_INT)	32	RW	0000_0000h
1_400Ch	Channel System Bus (CH5_SBR)	32	RW	0000_8002h
1_4010h	Channel Priority (CH5_PRI)	32	RW	0000_0000h
1_4020h	TCD Source Address (TCD5_SADDR)	32	RW	0000_0000h
1_4024h	TCD Signed Source Address Offset (TCD5_SOFF)	16	RW	0000h
1_4026h	TCD Transfer Attributes (TCD5_ATTR)	16	RW	0000h
1_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD5_NBYTES_MLOFFNO)	32	RW	0000_0000h
1_4028h	TCD Transfer Size with Minor Loop Offsets (TCD5_NBYTES_MLOFFYES)	32	RW	0000_0000h
1_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD5_SLAST_SDA)	32	RW	0000_0000h
1_4030h	TCD Destination Address (TCD5_DADDR)	32	RW	0000_0000h
1_4034h	TCD Signed Destination Address Offset (TCD5_DOFF)	16	RW	0000h
1_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD5_CITER_ELINKNO)	16	RW	0000h
1_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD5_CITER_ELINKYES)	16	RW	0000h
1_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD5_DLAST_SGA)	32	RW	0000_0000h
1_403Ch	TCD Control and Status (TCD5_CSR)	16	RW	0000h
1_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD5_BITER_ELINKNO)	16	RW	0000h
1_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD5_BITER_ELINKYES)	16	RW	0000h
1_8000h	Channel Control and Status (CH6_CSR)	32	RW	0000_0000h
1_8004h	Channel Error Status (CH6_ES)	32	RW	0000_0000h
1_8008h	Channel Interrupt Status (CH6_INT)	32	RW	0000_0000h
1_800Ch	Channel System Bus (CH6_SBR)	32	RW	0000_8002h
1_8010h	Channel Priority (CH6_PRI)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8020h	TCD Source Address (TCD6_SADDR)	32	RW	0000_0000h
1_8024h	TCD Signed Source Address Offset (TCD6_SOFF)	16	RW	0000h
1_8026h	TCD Transfer Attributes (TCD6_ATTR)	16	RW	0000h
1_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD6_NBYTES_MLOFFNO)	32	RW	0000_0000h
1_8028h	TCD Transfer Size with Minor Loop Offsets (TCD6_NBYTES_MLOFFYES)	32	RW	0000_0000h
1_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD6_SLAST_SDA)	32	RW	0000_0000h
1_8030h	TCD Destination Address (TCD6_DADDR)	32	RW	0000_0000h
1_8034h	TCD Signed Destination Address Offset (TCD6_DOFF)	16	RW	0000h
1_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD6_CITER_ELINKNO)	16	RW	0000h
1_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD6_CITER_ELINKYES)	16	RW	0000h
1_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD6_DLAST_SGA)	32	RW	0000_0000h
1_803Ch	TCD Control and Status (TCD6_CSR)	16	RW	0000h
1_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD6_BITER_ELINKNO)	16	RW	0000h
1_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD6_BITER_ELINKYES)	16	RW	0000h
1_C000h	Channel Control and Status (CH7_CSR)	32	RW	0000_0000h
1_C004h	Channel Error Status (CH7_ES)	32	RW	0000_0000h
1_C008h	Channel Interrupt Status (CH7_INT)	32	RW	0000_0000h
1_C00Ch	Channel System Bus (CH7_SBR)	32	RW	0000_8002h
1_C010h	Channel Priority (CH7_PRI)	32	RW	0000_0000h
1_C020h	TCD Source Address (TCD7_SADDR)	32	RW	0000_0000h
1_C024h	TCD Signed Source Address Offset (TCD7_SOFF)	16	RW	0000h
1_C026h	TCD Transfer Attributes (TCD7_ATTR)	16	RW	0000h
1_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD7_NBYTES_MLOFFNO)	32	RW	0000_0000h
1_C028h	TCD Transfer Size with Minor Loop Offsets (TCD7_NBYTES_MLOFFYES)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD7_SLAST_SDA)	32	RW	0000_0000h
1_C030h	TCD Destination Address (TCD7_DADDR)	32	RW	0000_0000h
1_C034h	TCD Signed Destination Address Offset (TCD7_DOFF)	16	RW	0000h
1_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD7_CITER_ELINKNO)	16	RW	0000h
1_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD7_CITER_ELINKYES)	16	RW	0000h
1_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD7_DLAST_SGA)	32	RW	0000_0000h
1_C03Ch	TCD Control and Status (TCD7_CSR)	16	RW	0000h
1_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD7_BITER_ELINKNO)	16	RW	0000h
1_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD7_BITER_ELINKYES)	16	RW	0000h
2_0000h	Channel Control and Status (CH8_CSR)	32	RW	0000_0000h
2_0004h	Channel Error Status (CH8_ES)	32	RW	0000_0000h
2_0008h	Channel Interrupt Status (CH8_INT)	32	RW	0000_0000h
2_000Ch	Channel System Bus (CH8_SBR)	32	RW	0000_8002h
2_0010h	Channel Priority (CH8_PRI)	32	RW	0000_0000h
2_0020h	TCD Source Address (TCD8_SADDR)	32	RW	0000_0000h
2_0024h	TCD Signed Source Address Offset (TCD8_SOFF)	16	RW	0000h
2_0026h	TCD Transfer Attributes (TCD8_ATTR)	16	RW	0000h
2_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD8_NBYTES_MLOFFNO)	32	RW	0000_0000h
2_0028h	TCD Transfer Size with Minor Loop Offsets (TCD8_NBYTES_MLOFFYES)	32	RW	0000_0000h
2_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD8_SLAST_SDA)	32	RW	0000_0000h
2_0030h	TCD Destination Address (TCD8_DADDR)	32	RW	0000_0000h
2_0034h	TCD Signed Destination Address Offset (TCD8_DOFF)	16	RW	0000h
2_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD8_CITER_ELINKNO)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD8_CITER_ELINKYES)	16	RW	0000h
2_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD8_DLAST_SGA)	32	RW	0000_0000h
2_003Ch	TCD Control and Status (TCD8_CSR)	16	RW	0000h
2_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD8_BITER_ELINKNO)	16	RW	0000h
2_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD8_BITER_ELINKYES)	16	RW	0000h
2_4000h	Channel Control and Status (CH9_CSR)	32	RW	0000_0000h
2_4004h	Channel Error Status (CH9_ES)	32	RW	0000_0000h
2_4008h	Channel Interrupt Status (CH9_INT)	32	RW	0000_0000h
2_400Ch	Channel System Bus (CH9_SBR)	32	RW	0000_8002h
2_4010h	Channel Priority (CH9_PRI)	32	RW	0000_0000h
2_4020h	TCD Source Address (TCD9_SADDR)	32	RW	0000_0000h
2_4024h	TCD Signed Source Address Offset (TCD9_SOFF)	16	RW	0000h
2_4026h	TCD Transfer Attributes (TCD9_ATTR)	16	RW	0000h
2_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD9_NBYTES_MLOFFNO)	32	RW	0000_0000h
2_4028h	TCD Transfer Size with Minor Loop Offsets (TCD9_NBYTES_MLOFFYES)	32	RW	0000_0000h
2_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD9_SLAST_SDA)	32	RW	0000_0000h
2_4030h	TCD Destination Address (TCD9_DADDR)	32	RW	0000_0000h
2_4034h	TCD Signed Destination Address Offset (TCD9_DOFF)	16	RW	0000h
2_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD9_CITER_ELINKNO)	16	RW	0000h
2_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD9_CITER_ELINKYES)	16	RW	0000h
2_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD9_DLAST_SGA)	32	RW	0000_0000h
2_403Ch	TCD Control and Status (TCD9_CSR)	16	RW	0000h
2_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD9_BITER_ELINKNO)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD9_BITER_ELINKYES)	16	RW	0000h
2_8000h	Channel Control and Status (CH10_CSR)	32	RW	0000_0000h
2_8004h	Channel Error Status (CH10_ES)	32	RW	0000_0000h
2_8008h	Channel Interrupt Status (CH10_INT)	32	RW	0000_0000h
2_800Ch	Channel System Bus (CH10_SBR)	32	RW	0000_8002h
2_8010h	Channel Priority (CH10_PRI)	32	RW	0000_0000h
2_8020h	TCD Source Address (TCD10_SADDR)	32	RW	0000_0000h
2_8024h	TCD Signed Source Address Offset (TCD10_SOFF)	16	RW	0000h
2_8026h	TCD Transfer Attributes (TCD10_ATTR)	16	RW	0000h
2_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD10_NBYTES_MLOFFNO)	32	RW	0000_0000h
2_8028h	TCD Transfer Size with Minor Loop Offsets (TCD10_NBYTES_MLOFFYES)	32	RW	0000_0000h
2_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD10_SLAST_SDA)	32	RW	0000_0000h
2_8030h	TCD Destination Address (TCD10_DADDR)	32	RW	0000_0000h
2_8034h	TCD Signed Destination Address Offset (TCD10_DOFF)	16	RW	0000h
2_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD10_CITER_ELINKNO)	16	RW	0000h
2_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD10_CITER_ELINKYES)	16	RW	0000h
2_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD10_DLAST_SGA)	32	RW	0000_0000h
2_803Ch	TCD Control and Status (TCD10_CSR)	16	RW	0000h
2_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD10_BITER_ELINKNO)	16	RW	0000h
2_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD10_BITER_ELINKYES)	16	RW	0000h
2_C000h	Channel Control and Status (CH11_CSR)	32	RW	0000_0000h
2_C004h	Channel Error Status (CH11_ES)	32	RW	0000_0000h
2_C008h	Channel Interrupt Status (CH11_INT)	32	RW	0000_0000h
2_C00Ch	Channel System Bus (CH11_SBR)	32	RW	0000_8002h
2_C010h	Channel Priority (CH11_PRI)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2_C020h	TCD Source Address (TCD11_SADDR)	32	RW	0000_0000h
2_C024h	TCD Signed Source Address Offset (TCD11_SOFF)	16	RW	0000h
2_C026h	TCD Transfer Attributes (TCD11_ATTR)	16	RW	0000h
2_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD11_NBYTES_MLOFFNO)	32	RW	0000_0000h
2_C028h	TCD Transfer Size with Minor Loop Offsets (TCD11_NBYTES_MLOFFYES)	32	RW	0000_0000h
2_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD11_SLAST_SDA)	32	RW	0000_0000h
2_C030h	TCD Destination Address (TCD11_DADDR)	32	RW	0000_0000h
2_C034h	TCD Signed Destination Address Offset (TCD11_DOFF)	16	RW	0000h
2_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD11_CITER_ELINKNO)	16	RW	0000h
2_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD11_CITER_ELINKYES)	16	RW	0000h
2_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD11_DLAST_SGA)	32	RW	0000_0000h
2_C03Ch	TCD Control and Status (TCD11_CSR)	16	RW	0000h
2_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD11_BITER_ELINKNO)	16	RW	0000h
2_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD11_BITER_ELINKYES)	16	RW	0000h
20_0000h	Channel Control and Status (CH12_CSR)	32	RW	0000_0000h
20_0004h	Channel Error Status (CH12_ES)	32	RW	0000_0000h
20_0008h	Channel Interrupt Status (CH12_INT)	32	RW	0000_0000h
20_000Ch	Channel System Bus (CH12_SBR)	32	RW	0000_8002h
20_0010h	Channel Priority (CH12_PRI)	32	RW	0000_0000h
20_0020h	TCD Source Address (TCD12_SADDR)	32	RW	0000_0000h
20_0024h	TCD Signed Source Address Offset (TCD12_SOFF)	16	RW	0000h
20_0026h	TCD Transfer Attributes (TCD12_ATTR)	16	RW	0000h
20_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD12_NBYTES_MLOFFNO)	32	RW	0000_0000h
20_0028h	TCD Transfer Size with Minor Loop Offsets (TCD12_NBYTES_MLOFFYES)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD12_SLAST_SDA)	32	RW	0000_0000h
20_0030h	TCD Destination Address (TCD12_DADDR)	32	RW	0000_0000h
20_0034h	TCD Signed Destination Address Offset (TCD12_DOFF)	16	RW	0000h
20_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD12_CITER_ELINKNO)	16	RW	0000h
20_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD12_CITER_ELINKYES)	16	RW	0000h
20_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD12_DLAST_SGA)	32	RW	0000_0000h
20_003Ch	TCD Control and Status (TCD12_CSR)	16	RW	0000h
20_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD12_BITER_ELINKNO)	16	RW	0000h
20_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD12_BITER_ELINKYES)	16	RW	0000h
20_4000h	Channel Control and Status (CH13_CSR)	32	RW	0000_0000h
20_4004h	Channel Error Status (CH13_ES)	32	RW	0000_0000h
20_4008h	Channel Interrupt Status (CH13_INT)	32	RW	0000_0000h
20_400Ch	Channel System Bus (CH13_SBR)	32	RW	0000_8002h
20_4010h	Channel Priority (CH13_PRI)	32	RW	0000_0000h
20_4020h	TCD Source Address (TCD13_SADDR)	32	RW	0000_0000h
20_4024h	TCD Signed Source Address Offset (TCD13_SOFF)	16	RW	0000h
20_4026h	TCD Transfer Attributes (TCD13_ATTR)	16	RW	0000h
20_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD13_NBYTES_MLOFFNO)	32	RW	0000_0000h
20_4028h	TCD Transfer Size with Minor Loop Offsets (TCD13_NBYTES_MLOFFYES)	32	RW	0000_0000h
20_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD13_SLAST_SDA)	32	RW	0000_0000h
20_4030h	TCD Destination Address (TCD13_DADDR)	32	RW	0000_0000h
20_4034h	TCD Signed Destination Address Offset (TCD13_DOFF)	16	RW	0000h
20_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD13_CITER_ELINKNO)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD13_CITER_ELINKYES)	16	RW	0000h
20_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD13_DLAST_SGA)	32	RW	0000_0000h
20_403Ch	TCD Control and Status (TCD13_CSR)	16	RW	0000h
20_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD13_BITER_ELINKNO)	16	RW	0000h
20_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD13_BITER_ELINKYES)	16	RW	0000h
20_8000h	Channel Control and Status (CH14_CSR)	32	RW	0000_0000h
20_8004h	Channel Error Status (CH14_ES)	32	RW	0000_0000h
20_8008h	Channel Interrupt Status (CH14_INT)	32	RW	0000_0000h
20_800Ch	Channel System Bus (CH14_SBR)	32	RW	0000_8002h
20_8010h	Channel Priority (CH14_PRI)	32	RW	0000_0000h
20_8020h	TCD Source Address (TCD14_SADDR)	32	RW	0000_0000h
20_8024h	TCD Signed Source Address Offset (TCD14_SOFF)	16	RW	0000h
20_8026h	TCD Transfer Attributes (TCD14_ATTR)	16	RW	0000h
20_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD14_NBYTES_MLOFFNO)	32	RW	0000_0000h
20_8028h	TCD Transfer Size with Minor Loop Offsets (TCD14_NBYTES_MLOFFYES)	32	RW	0000_0000h
20_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD14_SLAST_SDA)	32	RW	0000_0000h
20_8030h	TCD Destination Address (TCD14_DADDR)	32	RW	0000_0000h
20_8034h	TCD Signed Destination Address Offset (TCD14_DOFF)	16	RW	0000h
20_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD14_CITER_ELINKNO)	16	RW	0000h
20_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD14_CITER_ELINKYES)	16	RW	0000h
20_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD14_DLAST_SGA)	32	RW	0000_0000h
20_803Ch	TCD Control and Status (TCD14_CSR)	16	RW	0000h
20_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD14_BITER_ELINKNO)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD14_BITER_ELINKYES)	16	RW	0000h
20_C000h	Channel Control and Status (CH15_CSR)	32	RW	0000_0000h
20_C004h	Channel Error Status (CH15_ES)	32	RW	0000_0000h
20_C008h	Channel Interrupt Status (CH15_INT)	32	RW	0000_0000h
20_C00Ch	Channel System Bus (CH15_SBR)	32	RW	0000_8002h
20_C010h	Channel Priority (CH15_PRI)	32	RW	0000_0000h
20_C020h	TCD Source Address (TCD15_SADDR)	32	RW	0000_0000h
20_C024h	TCD Signed Source Address Offset (TCD15_SOFF)	16	RW	0000h
20_C026h	TCD Transfer Attributes (TCD15_ATTR)	16	RW	0000h
20_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD15_NBYTES_MLOFFNO)	32	RW	0000_0000h
20_C028h	TCD Transfer Size with Minor Loop Offsets (TCD15_NBYTES_MLOFFYES)	32	RW	0000_0000h
20_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD15_SLAST_SDA)	32	RW	0000_0000h
20_C030h	TCD Destination Address (TCD15_DADDR)	32	RW	0000_0000h
20_C034h	TCD Signed Destination Address Offset (TCD15_DOFF)	16	RW	0000h
20_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD15_CITER_ELINKNO)	16	RW	0000h
20_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD15_CITER_ELINKYES)	16	RW	0000h
20_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD15_DLAST_SGA)	32	RW	0000_0000h
20_C03Ch	TCD Control and Status (TCD15_CSR)	16	RW	0000h
20_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD15_BITER_ELINKNO)	16	RW	0000h
20_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD15_BITER_ELINKYES)	16	RW	0000h
21_0000h	Channel Control and Status (CH16_CSR)	32	RW	0000_0000h
21_0004h	Channel Error Status (CH16_ES)	32	RW	0000_0000h
21_0008h	Channel Interrupt Status (CH16_INT)	32	RW	0000_0000h
21_000Ch	Channel System Bus (CH16_SBR)	32	RW	0000_8002h
21_0010h	Channel Priority (CH16_PRI)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
21_0020h	TCD Source Address (TCD16_SADDR)	32	RW	0000_0000h
21_0024h	TCD Signed Source Address Offset (TCD16_SOFF)	16	RW	0000h
21_0026h	TCD Transfer Attributes (TCD16_ATTR)	16	RW	0000h
21_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD16_NBYTES_MLOFFNO)	32	RW	0000_0000h
21_0028h	TCD Transfer Size with Minor Loop Offsets (TCD16_NBYTES_MLOFFYES)	32	RW	0000_0000h
21_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD16_SLAST_SDA)	32	RW	0000_0000h
21_0030h	TCD Destination Address (TCD16_DADDR)	32	RW	0000_0000h
21_0034h	TCD Signed Destination Address Offset (TCD16_DOFF)	16	RW	0000h
21_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD16_CITER_ELINKNO)	16	RW	0000h
21_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD16_CITER_ELINKYES)	16	RW	0000h
21_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD16_DLAST_SGA)	32	RW	0000_0000h
21_003Ch	TCD Control and Status (TCD16_CSR)	16	RW	0000h
21_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD16_BITER_ELINKNO)	16	RW	0000h
21_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD16_BITER_ELINKYES)	16	RW	0000h
21_4000h	Channel Control and Status (CH17_CSR)	32	RW	0000_0000h
21_4004h	Channel Error Status (CH17_ES)	32	RW	0000_0000h
21_4008h	Channel Interrupt Status (CH17_INT)	32	RW	0000_0000h
21_400Ch	Channel System Bus (CH17_SBR)	32	RW	0000_8002h
21_4010h	Channel Priority (CH17_PRI)	32	RW	0000_0000h
21_4020h	TCD Source Address (TCD17_SADDR)	32	RW	0000_0000h
21_4024h	TCD Signed Source Address Offset (TCD17_SOFF)	16	RW	0000h
21_4026h	TCD Transfer Attributes (TCD17_ATTR)	16	RW	0000h
21_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD17_NBYTES_MLOFFNO)	32	RW	0000_0000h
21_4028h	TCD Transfer Size with Minor Loop Offsets (TCD17_NBYTES_MLOFFYES)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
21_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD17_SLAST_SDA)	32	RW	0000_0000h
21_4030h	TCD Destination Address (TCD17_DADDR)	32	RW	0000_0000h
21_4034h	TCD Signed Destination Address Offset (TCD17_DOFF)	16	RW	0000h
21_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD17_CITER_ELINKNO)	16	RW	0000h
21_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD17_CITER_ELINKYES)	16	RW	0000h
21_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD17_DLAST_SGA)	32	RW	0000_0000h
21_403Ch	TCD Control and Status (TCD17_CSR)	16	RW	0000h
21_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD17_BITER_ELINKNO)	16	RW	0000h
21_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD17_BITER_ELINKYES)	16	RW	0000h
21_8000h	Channel Control and Status (CH18_CSR)	32	RW	0000_0000h
21_8004h	Channel Error Status (CH18_ES)	32	RW	0000_0000h
21_8008h	Channel Interrupt Status (CH18_INT)	32	RW	0000_0000h
21_800Ch	Channel System Bus (CH18_SBR)	32	RW	0000_8002h
21_8010h	Channel Priority (CH18_PRI)	32	RW	0000_0000h
21_8020h	TCD Source Address (TCD18_SADDR)	32	RW	0000_0000h
21_8024h	TCD Signed Source Address Offset (TCD18_SOFF)	16	RW	0000h
21_8026h	TCD Transfer Attributes (TCD18_ATTR)	16	RW	0000h
21_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD18_NBYTES_MLOFFNO)	32	RW	0000_0000h
21_8028h	TCD Transfer Size with Minor Loop Offsets (TCD18_NBYTES_MLOFFYES)	32	RW	0000_0000h
21_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD18_SLAST_SDA)	32	RW	0000_0000h
21_8030h	TCD Destination Address (TCD18_DADDR)	32	RW	0000_0000h
21_8034h	TCD Signed Destination Address Offset (TCD18_DOFF)	16	RW	0000h
21_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD18_CITER_ELINKNO)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
21_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD18_CITER_ELINKYES)	16	RW	0000h
21_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD18_DLAST_SGA)	32	RW	0000_0000h
21_803Ch	TCD Control and Status (TCD18_CSR)	16	RW	0000h
21_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD18_BITER_ELINKNO)	16	RW	0000h
21_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD18_BITER_ELINKYES)	16	RW	0000h
21_C000h	Channel Control and Status (CH19_CSR)	32	RW	0000_0000h
21_C004h	Channel Error Status (CH19_ES)	32	RW	0000_0000h
21_C008h	Channel Interrupt Status (CH19_INT)	32	RW	0000_0000h
21_C00Ch	Channel System Bus (CH19_SBR)	32	RW	0000_8002h
21_C010h	Channel Priority (CH19_PRI)	32	RW	0000_0000h
21_C020h	TCD Source Address (TCD19_SADDR)	32	RW	0000_0000h
21_C024h	TCD Signed Source Address Offset (TCD19_SOFF)	16	RW	0000h
21_C026h	TCD Transfer Attributes (TCD19_ATTR)	16	RW	0000h
21_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD19_NBYTES_MLOFFNO)	32	RW	0000_0000h
21_C028h	TCD Transfer Size with Minor Loop Offsets (TCD19_NBYTES_MLOFFYES)	32	RW	0000_0000h
21_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD19_SLAST_SDA)	32	RW	0000_0000h
21_C030h	TCD Destination Address (TCD19_DADDR)	32	RW	0000_0000h
21_C034h	TCD Signed Destination Address Offset (TCD19_DOFF)	16	RW	0000h
21_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD19_CITER_ELINKNO)	16	RW	0000h
21_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD19_CITER_ELINKYES)	16	RW	0000h
21_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD19_DLAST_SGA)	32	RW	0000_0000h
21_C03Ch	TCD Control and Status (TCD19_CSR)	16	RW	0000h
21_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD19_BITER_ELINKNO)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
21_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD19_BITER_ELINKYES)	16	RW	0000h
22_0000h	Channel Control and Status (CH20_CSR)	32	RW	0000_0000h
22_0004h	Channel Error Status (CH20_ES)	32	RW	0000_0000h
22_0008h	Channel Interrupt Status (CH20_INT)	32	RW	0000_0000h
22_000Ch	Channel System Bus (CH20_SBR)	32	RW	0000_8002h
22_0010h	Channel Priority (CH20_PRI)	32	RW	0000_0000h
22_0020h	TCD Source Address (TCD20_SADDR)	32	RW	0000_0000h
22_0024h	TCD Signed Source Address Offset (TCD20_SOFF)	16	RW	0000h
22_0026h	TCD Transfer Attributes (TCD20_ATTR)	16	RW	0000h
22_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD20_NBYTES_MLOFFNO)	32	RW	0000_0000h
22_0028h	TCD Transfer Size with Minor Loop Offsets (TCD20_NBYTES_MLOFFYES)	32	RW	0000_0000h
22_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD20_SLAST_SDA)	32	RW	0000_0000h
22_0030h	TCD Destination Address (TCD20_DADDR)	32	RW	0000_0000h
22_0034h	TCD Signed Destination Address Offset (TCD20_DOFF)	16	RW	0000h
22_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD20_CITER_ELINKNO)	16	RW	0000h
22_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD20_CITER_ELINKYES)	16	RW	0000h
22_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD20_DLAST_SGA)	32	RW	0000_0000h
22_003Ch	TCD Control and Status (TCD20_CSR)	16	RW	0000h
22_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD20_BITER_ELINKNO)	16	RW	0000h
22_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD20_BITER_ELINKYES)	16	RW	0000h
22_4000h	Channel Control and Status (CH21_CSR)	32	RW	0000_0000h
22_4004h	Channel Error Status (CH21_ES)	32	RW	0000_0000h
22_4008h	Channel Interrupt Status (CH21_INT)	32	RW	0000_0000h
22_400Ch	Channel System Bus (CH21_SBR)	32	RW	0000_8002h
22_4010h	Channel Priority (CH21_PRI)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
22_4020h	TCD Source Address (TCD21_SADDR)	32	RW	0000_0000h
22_4024h	TCD Signed Source Address Offset (TCD21_SOFF)	16	RW	0000h
22_4026h	TCD Transfer Attributes (TCD21_ATTR)	16	RW	0000h
22_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD21_NBYTES_MLOFFNO)	32	RW	0000_0000h
22_4028h	TCD Transfer Size with Minor Loop Offsets (TCD21_NBYTES_MLOFFYES)	32	RW	0000_0000h
22_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD21_SLAST_SDA)	32	RW	0000_0000h
22_4030h	TCD Destination Address (TCD21_DADDR)	32	RW	0000_0000h
22_4034h	TCD Signed Destination Address Offset (TCD21_DOFF)	16	RW	0000h
22_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD21_CITER_ELINKNO)	16	RW	0000h
22_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD21_CITER_ELINKYES)	16	RW	0000h
22_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD21_DLAST_SGA)	32	RW	0000_0000h
22_403Ch	TCD Control and Status (TCD21_CSR)	16	RW	0000h
22_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD21_BITER_ELINKNO)	16	RW	0000h
22_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD21_BITER_ELINKYES)	16	RW	0000h
22_8000h	Channel Control and Status (CH22_CSR)	32	RW	0000_0000h
22_8004h	Channel Error Status (CH22_ES)	32	RW	0000_0000h
22_8008h	Channel Interrupt Status (CH22_INT)	32	RW	0000_0000h
22_800Ch	Channel System Bus (CH22_SBR)	32	RW	0000_8002h
22_8010h	Channel Priority (CH22_PRI)	32	RW	0000_0000h
22_8020h	TCD Source Address (TCD22_SADDR)	32	RW	0000_0000h
22_8024h	TCD Signed Source Address Offset (TCD22_SOFF)	16	RW	0000h
22_8026h	TCD Transfer Attributes (TCD22_ATTR)	16	RW	0000h
22_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD22_NBYTES_MLOFFNO)	32	RW	0000_0000h
22_8028h	TCD Transfer Size with Minor Loop Offsets (TCD22_NBYTES_MLOFFYES)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
22_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD22_SLAST_SDA)	32	RW	0000_0000h
22_8030h	TCD Destination Address (TCD22_DADDR)	32	RW	0000_0000h
22_8034h	TCD Signed Destination Address Offset (TCD22_DOFF)	16	RW	0000h
22_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD22_CITER_ELINKNO)	16	RW	0000h
22_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD22_CITER_ELINKYES)	16	RW	0000h
22_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD22_DLAST_SGA)	32	RW	0000_0000h
22_803Ch	TCD Control and Status (TCD22_CSR)	16	RW	0000h
22_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD22_BITER_ELINKNO)	16	RW	0000h
22_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD22_BITER_ELINKYES)	16	RW	0000h
22_C000h	Channel Control and Status (CH23_CSR)	32	RW	0000_0000h
22_C004h	Channel Error Status (CH23_ES)	32	RW	0000_0000h
22_C008h	Channel Interrupt Status (CH23_INT)	32	RW	0000_0000h
22_C00Ch	Channel System Bus (CH23_SBR)	32	RW	0000_8002h
22_C010h	Channel Priority (CH23_PRI)	32	RW	0000_0000h
22_C020h	TCD Source Address (TCD23_SADDR)	32	RW	0000_0000h
22_C024h	TCD Signed Source Address Offset (TCD23_SOFF)	16	RW	0000h
22_C026h	TCD Transfer Attributes (TCD23_ATTR)	16	RW	0000h
22_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD23_NBYTES_MLOFFNO)	32	RW	0000_0000h
22_C028h	TCD Transfer Size with Minor Loop Offsets (TCD23_NBYTES_MLOFFYES)	32	RW	0000_0000h
22_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD23_SLAST_SDA)	32	RW	0000_0000h
22_C030h	TCD Destination Address (TCD23_DADDR)	32	RW	0000_0000h
22_C034h	TCD Signed Destination Address Offset (TCD23_DOFF)	16	RW	0000h
22_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD23_CITER_ELINKNO)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
22_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD23_CITER_ELINKYES)	16	RW	0000h
22_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD23_DLAST_SGA)	32	RW	0000_0000h
22_C03Ch	TCD Control and Status (TCD23_CSR)	16	RW	0000h
22_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD23_BITER_ELINKNO)	16	RW	0000h
22_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD23_BITER_ELINKYES)	16	RW	0000h
23_0000h	Channel Control and Status (CH24_CSR)	32	RW	0000_0000h
23_0004h	Channel Error Status (CH24_ES)	32	RW	0000_0000h
23_0008h	Channel Interrupt Status (CH24_INT)	32	RW	0000_0000h
23_000Ch	Channel System Bus (CH24_SBR)	32	RW	0000_8002h
23_0010h	Channel Priority (CH24_PRI)	32	RW	0000_0000h
23_0020h	TCD Source Address (TCD24_SADDR)	32	RW	0000_0000h
23_0024h	TCD Signed Source Address Offset (TCD24_SOFF)	16	RW	0000h
23_0026h	TCD Transfer Attributes (TCD24_ATTR)	16	RW	0000h
23_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD24_NBYTES_MLOFFNO)	32	RW	0000_0000h
23_0028h	TCD Transfer Size with Minor Loop Offsets (TCD24_NBYTES_MLOFFYES)	32	RW	0000_0000h
23_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD24_SLAST_SDA)	32	RW	0000_0000h
23_0030h	TCD Destination Address (TCD24_DADDR)	32	RW	0000_0000h
23_0034h	TCD Signed Destination Address Offset (TCD24_DOFF)	16	RW	0000h
23_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD24_CITER_ELINKNO)	16	RW	0000h
23_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD24_CITER_ELINKYES)	16	RW	0000h
23_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD24_DLAST_SGA)	32	RW	0000_0000h
23_003Ch	TCD Control and Status (TCD24_CSR)	16	RW	0000h
23_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD24_BITER_ELINKNO)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
23_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD24_BITER_ELINKYES)	16	RW	0000h
23_4000h	Channel Control and Status (CH25_CSR)	32	RW	0000_0000h
23_4004h	Channel Error Status (CH25_ES)	32	RW	0000_0000h
23_4008h	Channel Interrupt Status (CH25_INT)	32	RW	0000_0000h
23_400Ch	Channel System Bus (CH25_SBR)	32	RW	0000_8002h
23_4010h	Channel Priority (CH25_PRI)	32	RW	0000_0000h
23_4020h	TCD Source Address (TCD25_SADDR)	32	RW	0000_0000h
23_4024h	TCD Signed Source Address Offset (TCD25_SOFF)	16	RW	0000h
23_4026h	TCD Transfer Attributes (TCD25_ATTR)	16	RW	0000h
23_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD25_NBYTES_MLOFFNO)	32	RW	0000_0000h
23_4028h	TCD Transfer Size with Minor Loop Offsets (TCD25_NBYTES_MLOFFYES)	32	RW	0000_0000h
23_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD25_SLAST_SDA)	32	RW	0000_0000h
23_4030h	TCD Destination Address (TCD25_DADDR)	32	RW	0000_0000h
23_4034h	TCD Signed Destination Address Offset (TCD25_DOFF)	16	RW	0000h
23_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD25_CITER_ELINKNO)	16	RW	0000h
23_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD25_CITER_ELINKYES)	16	RW	0000h
23_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD25_DLAST_SGA)	32	RW	0000_0000h
23_403Ch	TCD Control and Status (TCD25_CSR)	16	RW	0000h
23_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD25_BITER_ELINKNO)	16	RW	0000h
23_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD25_BITER_ELINKYES)	16	RW	0000h
23_8000h	Channel Control and Status (CH26_CSR)	32	RW	0000_0000h
23_8004h	Channel Error Status (CH26_ES)	32	RW	0000_0000h
23_8008h	Channel Interrupt Status (CH26_INT)	32	RW	0000_0000h
23_800Ch	Channel System Bus (CH26_SBR)	32	RW	0000_8002h
23_8010h	Channel Priority (CH26_PRI)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
23_8020h	TCD Source Address (TCD26_SADDR)	32	RW	0000_0000h
23_8024h	TCD Signed Source Address Offset (TCD26_SOFF)	16	RW	0000h
23_8026h	TCD Transfer Attributes (TCD26_ATTR)	16	RW	0000h
23_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD26_NBYTES_MLOFFNO)	32	RW	0000_0000h
23_8028h	TCD Transfer Size with Minor Loop Offsets (TCD26_NBYTES_MLOFFYES)	32	RW	0000_0000h
23_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD26_SLAST_SDA)	32	RW	0000_0000h
23_8030h	TCD Destination Address (TCD26_DADDR)	32	RW	0000_0000h
23_8034h	TCD Signed Destination Address Offset (TCD26_DOFF)	16	RW	0000h
23_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD26_CITER_ELINKNO)	16	RW	0000h
23_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD26_CITER_ELINKYES)	16	RW	0000h
23_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD26_DLAST_SGA)	32	RW	0000_0000h
23_803Ch	TCD Control and Status (TCD26_CSR)	16	RW	0000h
23_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD26_BITER_ELINKNO)	16	RW	0000h
23_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD26_BITER_ELINKYES)	16	RW	0000h
23_C000h	Channel Control and Status (CH27_CSR)	32	RW	0000_0000h
23_C004h	Channel Error Status (CH27_ES)	32	RW	0000_0000h
23_C008h	Channel Interrupt Status (CH27_INT)	32	RW	0000_0000h
23_C00Ch	Channel System Bus (CH27_SBR)	32	RW	0000_8002h
23_C010h	Channel Priority (CH27_PRI)	32	RW	0000_0000h
23_C020h	TCD Source Address (TCD27_SADDR)	32	RW	0000_0000h
23_C024h	TCD Signed Source Address Offset (TCD27_SOFF)	16	RW	0000h
23_C026h	TCD Transfer Attributes (TCD27_ATTR)	16	RW	0000h
23_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD27_NBYTES_MLOFFNO)	32	RW	0000_0000h
23_C028h	TCD Transfer Size with Minor Loop Offsets (TCD27_NBYTES_MLOFFYES)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
23_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD27_SLAST_SDA)	32	RW	0000_0000h
23_C030h	TCD Destination Address (TCD27_DADDR)	32	RW	0000_0000h
23_C034h	TCD Signed Destination Address Offset (TCD27_DOFF)	16	RW	0000h
23_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD27_CITER_ELINKNO)	16	RW	0000h
23_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD27_CITER_ELINKYES)	16	RW	0000h
23_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD27_DLAST_SGA)	32	RW	0000_0000h
23_C03Ch	TCD Control and Status (TCD27_CSR)	16	RW	0000h
23_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD27_BITER_ELINKNO)	16	RW	0000h
23_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD27_BITER_ELINKYES)	16	RW	0000h
24_0000h	Channel Control and Status (CH28_CSR)	32	RW	0000_0000h
24_0004h	Channel Error Status (CH28_ES)	32	RW	0000_0000h
24_0008h	Channel Interrupt Status (CH28_INT)	32	RW	0000_0000h
24_000Ch	Channel System Bus (CH28_SBR)	32	RW	0000_8002h
24_0010h	Channel Priority (CH28_PRI)	32	RW	0000_0000h
24_0020h	TCD Source Address (TCD28_SADDR)	32	RW	0000_0000h
24_0024h	TCD Signed Source Address Offset (TCD28_SOFF)	16	RW	0000h
24_0026h	TCD Transfer Attributes (TCD28_ATTR)	16	RW	0000h
24_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD28_NBYTES_MLOFFNO)	32	RW	0000_0000h
24_0028h	TCD Transfer Size with Minor Loop Offsets (TCD28_NBYTES_MLOFFYES)	32	RW	0000_0000h
24_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD28_SLAST_SDA)	32	RW	0000_0000h
24_0030h	TCD Destination Address (TCD28_DADDR)	32	RW	0000_0000h
24_0034h	TCD Signed Destination Address Offset (TCD28_DOFF)	16	RW	0000h
24_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD28_CITER_ELINKNO)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
24_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD28_CITER_ELINKYES)	16	RW	0000h
24_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD28_DLAST_SGA)	32	RW	0000_0000h
24_003Ch	TCD Control and Status (TCD28_CSR)	16	RW	0000h
24_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD28_BITER_ELINKNO)	16	RW	0000h
24_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD28_BITER_ELINKYES)	16	RW	0000h
24_4000h	Channel Control and Status (CH29_CSR)	32	RW	0000_0000h
24_4004h	Channel Error Status (CH29_ES)	32	RW	0000_0000h
24_4008h	Channel Interrupt Status (CH29_INT)	32	RW	0000_0000h
24_400Ch	Channel System Bus (CH29_SBR)	32	RW	0000_8002h
24_4010h	Channel Priority (CH29_PRI)	32	RW	0000_0000h
24_4020h	TCD Source Address (TCD29_SADDR)	32	RW	0000_0000h
24_4024h	TCD Signed Source Address Offset (TCD29_SOFF)	16	RW	0000h
24_4026h	TCD Transfer Attributes (TCD29_ATTR)	16	RW	0000h
24_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD29_NBYTES_MLOFFNO)	32	RW	0000_0000h
24_4028h	TCD Transfer Size with Minor Loop Offsets (TCD29_NBYTES_MLOFFYES)	32	RW	0000_0000h
24_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD29_SLAST_SDA)	32	RW	0000_0000h
24_4030h	TCD Destination Address (TCD29_DADDR)	32	RW	0000_0000h
24_4034h	TCD Signed Destination Address Offset (TCD29_DOFF)	16	RW	0000h
24_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD29_CITER_ELINKNO)	16	RW	0000h
24_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD29_CITER_ELINKYES)	16	RW	0000h
24_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD29_DLAST_SGA)	32	RW	0000_0000h
24_403Ch	TCD Control and Status (TCD29_CSR)	16	RW	0000h
24_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD29_BITER_ELINKNO)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
24_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD29_BITER_ELINKYES)	16	RW	0000h
24_8000h	Channel Control and Status (CH30_CSR)	32	RW	0000_0000h
24_8004h	Channel Error Status (CH30_ES)	32	RW	0000_0000h
24_8008h	Channel Interrupt Status (CH30_INT)	32	RW	0000_0000h
24_800Ch	Channel System Bus (CH30_SBR)	32	RW	0000_8002h
24_8010h	Channel Priority (CH30_PRI)	32	RW	0000_0000h
24_8020h	TCD Source Address (TCD30_SADDR)	32	RW	0000_0000h
24_8024h	TCD Signed Source Address Offset (TCD30_SOFF)	16	RW	0000h
24_8026h	TCD Transfer Attributes (TCD30_ATTR)	16	RW	0000h
24_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD30_NBYTES_MLOFFNO)	32	RW	0000_0000h
24_8028h	TCD Transfer Size with Minor Loop Offsets (TCD30_NBYTES_MLOFFYES)	32	RW	0000_0000h
24_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD30_SLAST_SDA)	32	RW	0000_0000h
24_8030h	TCD Destination Address (TCD30_DADDR)	32	RW	0000_0000h
24_8034h	TCD Signed Destination Address Offset (TCD30_DOFF)	16	RW	0000h
24_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD30_CITER_ELINKNO)	16	RW	0000h
24_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD30_CITER_ELINKYES)	16	RW	0000h
24_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD30_DLAST_SGA)	32	RW	0000_0000h
24_803Ch	TCD Control and Status (TCD30_CSR)	16	RW	0000h
24_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD30_BITER_ELINKNO)	16	RW	0000h
24_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD30_BITER_ELINKYES)	16	RW	0000h
24_C000h	Channel Control and Status (CH31_CSR)	32	RW	0000_0000h
24_C004h	Channel Error Status (CH31_ES)	32	RW	0000_0000h
24_C008h	Channel Interrupt Status (CH31_INT)	32	RW	0000_0000h
24_C00Ch	Channel System Bus (CH31_SBR)	32	RW	0000_8002h
24_C010h	Channel Priority (CH31_PRI)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
24_C020h	TCD Source Address (TCD31_SADDR)	32	RW	0000_0000h
24_C024h	TCD Signed Source Address Offset (TCD31_SOFF)	16	RW	0000h
24_C026h	TCD Transfer Attributes (TCD31_ATTR)	16	RW	0000h
24_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD31_NBYTES_MLOFFNO)	32	RW	0000_0000h
24_C028h	TCD Transfer Size with Minor Loop Offsets (TCD31_NBYTES_MLOFFYES)	32	RW	0000_0000h
24_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD31_SLAST_SDA)	32	RW	0000_0000h
24_C030h	TCD Destination Address (TCD31_DADDR)	32	RW	0000_0000h
24_C034h	TCD Signed Destination Address Offset (TCD31_DOFF)	16	RW	0000h
24_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD31_CITER_ELINKNO)	16	RW	0000h
24_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD31_CITER_ELINKYES)	16	RW	0000h
24_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD31_DLAST_SGA)	32	RW	0000_0000h
24_C03Ch	TCD Control and Status (TCD31_CSR)	16	RW	0000h
24_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD31_BITER_ELINKNO)	16	RW	0000h
24_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD31_BITER_ELINKYES)	16	RW	0000h

14.6.2.2 Channel Control and Status (CH0_CSR - CH31_CSR)

Offset

Register	Offset
CH0_CSR	0h
CH1_CSR	4000h
CH2_CSR	8000h
CH3_CSR	C000h
CH4_CSR	1_0000h
CH5_CSR	1_4000h

Table continues on the next page...

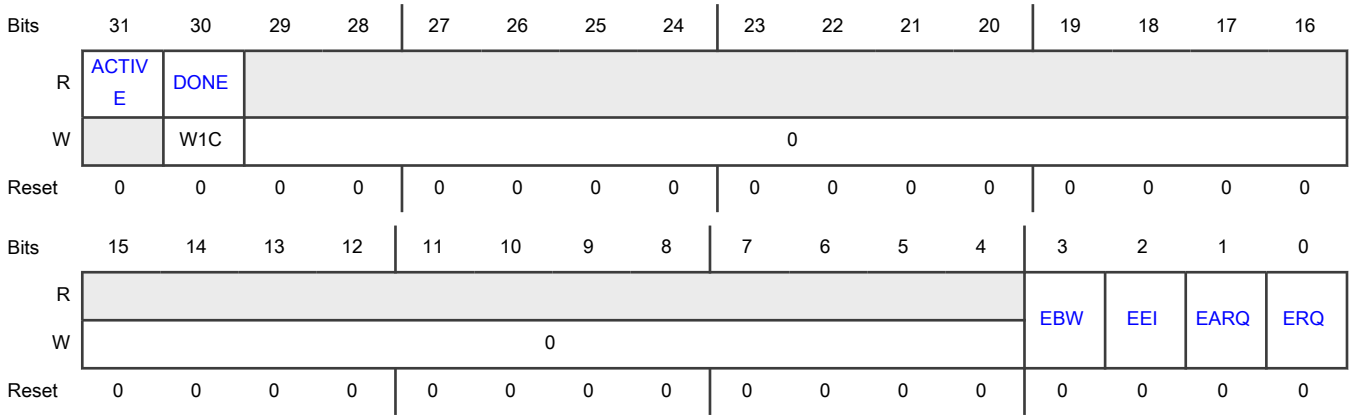
Table continued from the previous page...

Register	Offset
CH6_CSR	1_8000h
CH7_CSR	1_C000h
CH8_CSR	2_0000h
CH9_CSR	2_4000h
CH10_CSR	2_8000h
CH11_CSR	2_C000h
CH12_CSR	20_0000h
CH13_CSR	20_4000h
CH14_CSR	20_8000h
CH15_CSR	20_C000h
CH16_CSR	21_0000h
CH17_CSR	21_4000h
CH18_CSR	21_8000h
CH19_CSR	21_C000h
CH20_CSR	22_0000h
CH21_CSR	22_4000h
CH22_CSR	22_8000h
CH23_CSR	22_C000h
CH24_CSR	23_0000h
CH25_CSR	23_4000h
CH26_CSR	23_8000h
CH27_CSR	23_C000h
CH28_CSR	24_0000h
CH29_CSR	24_4000h
CH30_CSR	24_8000h
CH31_CSR	24_C000h

Function

This register contains several fields related to hardware and interrupt requests, configuration, and status for the given channel.

Diagram



Fields

Field	Function
31 ACTIVE	Channel Active The ACTIVE field indicates the channel was selected by arbitration and is executing the prescribed transfers. The eDMA sets it to 1 when channel service begins, and clears it to 0 as the minor loop completes or when any error condition is detected. Except for dynamic scatter/gather or dynamic channel linking, you must not modify the transfer control descriptor when a channel is active.
30 DONE	Channel Done The DONE field indicates the eDMA has completed the major loop. The eDMA engine sets this field as the CITER count reaches zero. If enabled, the eDMA generates an interrupt request corresponding to this completed channel. The software clears it, or the hardware clears it when the channel is activated. **NOTE** This field must be cleared to 0 before writing the MAJORELINK or ESG fields.
29-4 —	Reserved
3 EBW	Enable Buffered Writes When buffered writes are enabled, all writes except for the last write sequence of the minor loop are signaled by the eDMA as bufferable. 0b - Buffered writes on system bus disabled. Buffered writes on system bus disabled 1b - Buffered writes on system bus enabled. Bufferable write signal asserted on all system bus writes except during last write sequence
2 EEI	Enable Error Interrupt The EEI field enables the error interrupt signal for the channel. The DMA error indicator and the error interrupt enable flag must be asserted before an error interrupt request for a given channel is asserted to the interrupt controller. 0b - Error signal for corresponding channel does not generate error interrupt

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Assertion of error signal for corresponding channel generates error interrupt request
1 EARQ	Enable Asynchronous DMA Request The enable asynchronous DMA request field (EARQ) does not affect DMA operations. When set to 1, this field allows the hardware service request enable field (ERQ) to propagate out of the DMA to the power controller. When cleared to 0, this field masks the hardware service request enable field to the power controller. 0b - Disable asynchronous DMA request for the channel 1b - Enable asynchronous DMA request for the channel
0 ERQ	Enable DMA Request Disable a channel's hardware service request at the source before clearing the channel's ERQ field. The DMA hardware request input signal and the enable request field (ERQ) must be asserted before a channel's hardware service request is accepted. The state of the eDMA enable request field does not affect a channel service request made explicitly through software or channel linking. The state of the ERQ field does not affect the channel's START field. 0b - DMA hardware request signal for corresponding channel disabled 1b - DMA hardware request signal for corresponding channel enabled

14.6.2.3 Channel Error Status (CH0_ES - CH31_ES)

Offset

Register	Offset
CH0_ES	4h
CH1_ES	4004h
CH2_ES	8004h
CH3_ES	C004h
CH4_ES	1_0004h
CH5_ES	1_4004h
CH6_ES	1_8004h
CH7_ES	1_C004h
CH8_ES	2_0004h
CH9_ES	2_4004h
CH10_ES	2_8004h
CH11_ES	2_C004h
CH12_ES	20_0004h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
CH13_ES	20_4004h
CH14_ES	20_8004h
CH15_ES	20_C004h
CH16_ES	21_0004h
CH17_ES	21_4004h
CH18_ES	21_8004h
CH19_ES	21_C004h
CH20_ES	22_0004h
CH21_ES	22_4004h
CH22_ES	22_8004h
CH23_ES	22_C004h
CH24_ES	23_0004h
CH25_ES	23_4004h
CH26_ES	23_8004h
CH27_ES	23_C004h
CH28_ES	24_0004h
CH29_ES	24_4004h
CH30_ES	24_8004h
CH31_ES	24_C004h

Function

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- An illegal setting in the transfer control descriptor
- An error termination to a bus master read or write cycle

The ERR field signals the presence of an error for the channel. The eDMA engine signals the occurrence of an error condition by setting the appropriate field in this register. The outputs of this register are enabled by the contents of the CHn_CSR[EEI] field, then logically summed across all channels to form an error interrupt request, which may be routed to the interrupt controller. In addition, this enabled error status is logically OR'd onto the channel done interrupt, CHn_INT[INT], thus forming a done or error interrupt on a per channel basis.

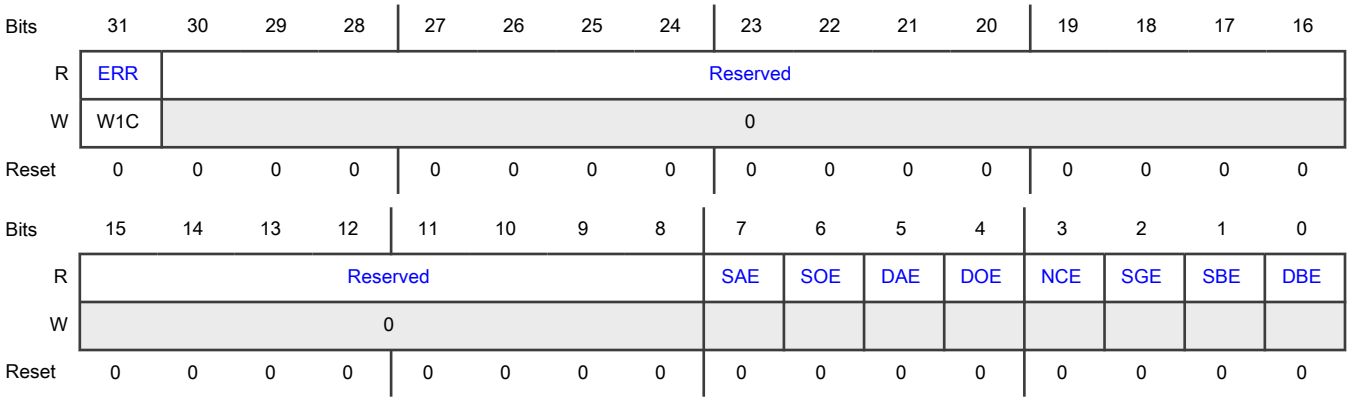
During the execution of the interrupt service routine associated with any DMA errors, it is software's responsibility to clear the appropriate bit, negating the error-interrupt request. The normal DMA channel completion indicators (setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request) are not affected when eDMA detects an error. The contents of this ERR register field can also be polled because a non-zero value indicates the presence of a channel error, regardless of the state of the EEI mask.

The state of any given channel's error indicators is affected by writes to this register. Writing a 1 to the ERR field clears the channel's error status, and writing a 0 has no effect.

An unspecified error, where only the ERR field is set to 1, indicates that either a transfer was cancelled with an error or else an uncorrectable TCD error occurred. The Management Page Error Status register has full view of the error condition.

See Fault reporting and handling for more details.

Diagram



Fields

Field	Function
31 ERR	Error In Channel 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
30-8 —	Reserved
7 SAE	Source Address Error TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SADDR field
6 SOE	Source Offset Error TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SOFF field
5 DAE	Destination Address Error TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DADDR field
4	Destination Offset Error TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE].

Table continues on the next page...

Table continued from the previous page...

Field	Function
DOE	0b - No destination offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DOFF field
3 NCE	NBYTES/CITER Configuration Error This error indicates that one of the following has occurred: • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE] • TCDn_CITER[CITER] is equal to zero • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK] 0b - No NBYTES/CITER configuration error 1b - Last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields
2 SGE	Scatter/Gather Configuration Error When this field is 1, it indicates that TCDn_DLAST_SGA is not on a 32-byte boundary. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG]is enabled. 0b - No scatter/gather configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DLAST_SGA field
1 SBE	Source Bus Error 0b - No source bus error 1b - Last recorded error was bus error on source read
0 DBE	Destination Bus Error 0b - No destination bus error 1b - Last recorded error was bus error on destination write

14.6.2.4 Channel Interrupt Status (CH0_INT - CH31_INT)

Offset

Register	Offset
CH0_INT	8h
CH1_INT	4008h
CH2_INT	8008h
CH3_INT	C008h
CH4_INT	1_0008h
CH5_INT	1_4008h

Table continues on the next page...

Table continued from the previous page...

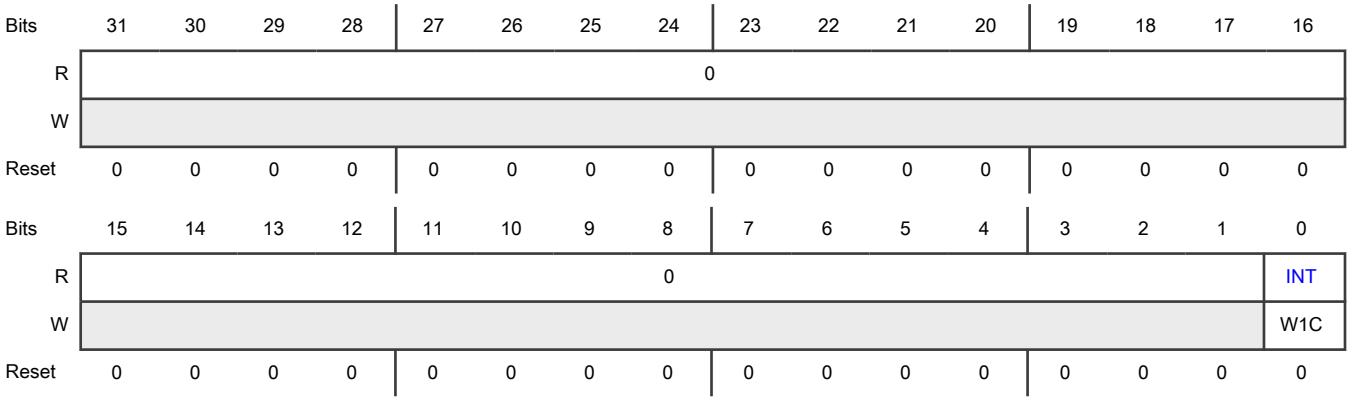
Register	Offset
CH6_INT	1_8008h
CH7_INT	1_C008h
CH8_INT	2_0008h
CH9_INT	2_4008h
CH10_INT	2_8008h
CH11_INT	2_C008h
CH12_INT	20_0008h
CH13_INT	20_4008h
CH14_INT	20_8008h
CH15_INT	20_C008h
CH16_INT	21_0008h
CH17_INT	21_4008h
CH18_INT	21_8008h
CH19_INT	21_C008h
CH20_INT	22_0008h
CH21_INT	22_4008h
CH22_INT	22_8008h
CH23_INT	22_C008h
CH24_INT	23_0008h
CH25_INT	23_4008h
CH26_INT	23_8008h
CH27_INT	23_C008h
CH28_INT	24_0008h
CH29_INT	24_4008h
CH30_INT	24_8008h
CH31_INT	24_C008h

Function

The INT field signals the presence of an interrupt request for the channel. Depending on the appropriate bit setting in the transfer control descriptors, the eDMA engine generates an interrupt on data transfer completion or an error condition.

The outputs of this register are directly routed to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit, negating the interrupt request. On writes to INT, a 1 clears the channel's interrupt request. A zero has no effect on the channel's current interrupt status.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 INT	Interrupt Request 0b - Interrupt request for corresponding channel cleared 1b - Interrupt request for corresponding channel active

14.6.2.5 Channel System Bus (CH0_SBR - CH31_SBR)

Offset

Register	Offset
CH0_SBR	Ch
CH1_SBR	400Ch
CH2_SBR	800Ch
CH3_SBR	C00Ch
CH4_SBR	1_000Ch
CH5_SBR	1_400Ch
CH6_SBR	1_800Ch
CH7_SBR	1_C00Ch
CH8_SBR	2_000Ch
CH9_SBR	2_400Ch
CH10_SBR	2_800Ch
CH11_SBR	2_C00Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
CH12_SBR	20_000Ch
CH13_SBR	20_400Ch
CH14_SBR	20_800Ch
CH15_SBR	20_C00Ch
CH16_SBR	21_000Ch
CH17_SBR	21_400Ch
CH18_SBR	21_800Ch
CH19_SBR	21_C00Ch
CH20_SBR	22_000Ch
CH21_SBR	22_400Ch
CH22_SBR	22_800Ch
CH23_SBR	22_C00Ch
CH24_SBR	23_000Ch
CH25_SBR	23_400Ch
CH26_SBR	23_800Ch
CH27_SBR	23_C00Ch
CH28_SBR	24_000Ch
CH29_SBR	24_400Ch
CH30_SBR	24_800Ch
CH31_SBR	24_C00Ch

Function

The Channel System Bus register places identification and attribute information on the system bus interface for the eDMA.

Diagram

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

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

Fields

Field	Function
31-20 —	Reserved
19-17 ATTR	Attribute Output DMA's system bus attribute output value.
16 EMI	Enable Master ID Replication The eDMA master ID replication field allows the eDMA to use the same protection level and system bus ID of the master programming the eDMA's TCD. When enabled, the eDMA uses the master ID and protection level stored in the CHn_SBR registers, instead of the eDMA's default values. When a master (for example a core) programs a TCD, its master ID is captured when the TCD*n*_CSR control attributes are written. A scatter/gather operation does not affect the CHn_SBR registers. You can write the EMI only if CSR[GMRC] = 1, which means Global Master ID Replication Control is enabled; otherwise, the EMI is forced to zero. **NOTE** If master ID replication is disabled, the privileged protection level (Supervisor mode) for DMA transfers is used. 0b - Master ID replication is disabled 1b - Master ID replication is enabled
15 PAL	Privileged Access Level This field controls DMA's protection level on the system bus when the channel is active. **NOTE** When you enable master ID replication, the value captured in this register is the privilege level of the core or other master writing the channel's transfer control descriptor, which is the lower byte of TCD*n*_CSR. 0b - User protection level for DMA transfers 1b - Privileged protection level for DMA transfers
14-4 —	Reserved
3-0 MID	Master ID This field controls the DMA's master ID on the system bus when the channel is active. **NOTE** The ID captured in this register reflects the master ID of the core or other master writing the channel's control attributes, which are in the lower byte of TCD*n*_CSR.

14.6.2.6 Channel Priority (CH0_PRI - CH31_PRI)

Offset

Register	Offset
CH0_PRI	10h
CH1_PRI	4010h
CH2_PRI	8010h
CH3_PRI	C010h
CH4_PRI	1_0010h
CH5_PRI	1_4010h
CH6_PRI	1_8010h
CH7_PRI	1_C010h
CH8_PRI	2_0010h
CH9_PRI	2_4010h
CH10_PRI	2_8010h
CH11_PRI	2_C010h
CH12_PRI	20_0010h
CH13_PRI	20_4010h
CH14_PRI	20_8010h
CH15_PRI	20_C010h
CH16_PRI	21_0010h
CH17_PRI	21_4010h
CH18_PRI	21_8010h
CH19_PRI	21_C010h
CH20_PRI	22_0010h
CH21_PRI	22_4010h
CH22_PRI	22_8010h
CH23_PRI	22_C010h
CH24_PRI	23_0010h
CH25_PRI	23_4010h
CH26_PRI	23_8010h
CH27_PRI	23_C010h
CH28_PRI	24_0010h
CH29_PRI	24_4010h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
CH30_PRI	24_8010h
CH31_PRI	24_C010h

Function

The contents of these registers define unique priorities associated with each channel within the same channel group. Channel grouping is programmed via Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI).

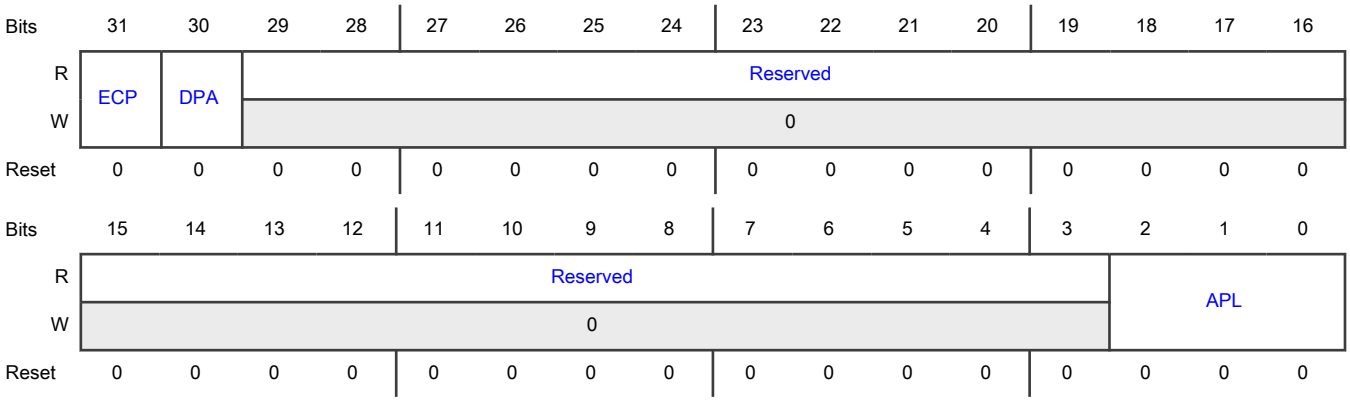
The channel priorities within a group are evaluated by numeric value; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, and so on. Software must program the channel priorities with unique values; otherwise, channel numbers with the same, non-zero value, will be selected based on channel number with the higher channel number having higher priority.

If more than one channel in a group has an arbitration priority level value of zero, then the arbitration mode field CSR[ERCA] is used to determine the arbitration scheme for all channels with APL=0 within a group.

When you enable round-robin channel arbitration (CSR[ERCA] = 1), all channels with APL=0 within a group will use a round-robin arbitration scheme, which rotates among these channels requesting service without regard to priority. Round-robin provides a fairness mechanism within an arbitration group.

When you enable fixed-priority channel arbitration (CSR[ERCA] = 0), eDMA selects channels with APL=0 based on channel number, with the higher channel number having higher priority.

Diagram



Fields

Field	Function
31 ECP	Enable Channel Preemption 0b - Channel cannot be suspended by a higher-priority channel's service request 1b - Channel can be temporarily suspended by a higher-priority channel's service request
30 DPA	Disable Preempt Ability 0b - Channel can suspend a lower-priority channel 1b - Channel cannot suspend any other channel, regardless of channel priority

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-3 —	Reserved
2-0 APL	Arbitration Priority Level Channel priority level for arbitration within the assigned arbitration group.

14.6.2.7 TCD Source Address (TCD0_SADDR - TCD31_SADDR)

Offset

Register	Offset
TCD0_SADDR	20h
TCD1_SADDR	4020h
TCD2_SADDR	8020h
TCD3_SADDR	C020h
TCD4_SADDR	1_0020h
TCD5_SADDR	1_4020h
TCD6_SADDR	1_8020h
TCD7_SADDR	1_C020h
TCD8_SADDR	2_0020h
TCD9_SADDR	2_4020h
TCD10_SADDR	2_8020h
TCD11_SADDR	2_C020h
TCD12_SADDR	20_0020h
TCD13_SADDR	20_4020h
TCD14_SADDR	20_8020h
TCD15_SADDR	20_C020h
TCD16_SADDR	21_0020h
TCD17_SADDR	21_4020h
TCD18_SADDR	21_8020h
TCD19_SADDR	21_C020h
TCD20_SADDR	22_0020h
TCD21_SADDR	22_4020h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD22_SADDR	22_8020h
TCD23_SADDR	22_C020h
TCD24_SADDR	23_0020h
TCD25_SADDR	23_4020h
TCD26_SADDR	23_8020h
TCD27_SADDR	23_C020h
TCD28_SADDR	24_0020h
TCD29_SADDR	24_4020h
TCD30_SADDR	24_8020h
TCD31_SADDR	24_C020h

Function

This register contains the address for the read transactions.

Diagram

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

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

Fields

Field	Function
31-0 SADDR	Source Address Memory address pointing to the source data.

14.6.2.8 TCD Signed Source Address Offset (TCD0_SOFF - TCD31_SOFF)

Offset

Register	Offset
TCD0_SOFF	24h
TCD1_SOFF	4024h
TCD2_SOFF	8024h
TCD3_SOFF	C024h
TCD4_SOFF	1_0024h
TCD5_SOFF	1_4024h
TCD6_SOFF	1_8024h
TCD7_SOFF	1_C024h
TCD8_SOFF	2_0024h
TCD9_SOFF	2_4024h
TCD10_SOFF	2_8024h
TCD11_SOFF	2_C024h
TCD12_SOFF	20_0024h
TCD13_SOFF	20_4024h
TCD14_SOFF	20_8024h
TCD15_SOFF	20_C024h
TCD16_SOFF	21_0024h
TCD17_SOFF	21_4024h
TCD18_SOFF	21_8024h
TCD19_SOFF	21_C024h
TCD20_SOFF	22_0024h
TCD21_SOFF	22_4024h
TCD22_SOFF	22_8024h
TCD23_SOFF	22_C024h
TCD24_SOFF	23_0024h
TCD25_SOFF	23_4024h
TCD26_SOFF	23_8024h
TCD27_SOFF	23_C024h
TCD28_SOFF	24_0024h
TCD29_SOFF	24_4024h

Table continues on the next page...

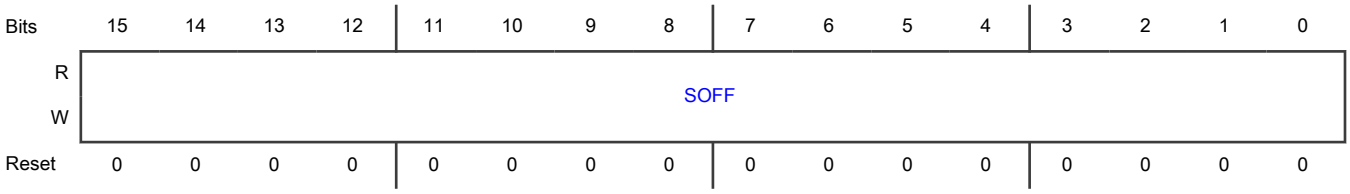
Table continued from the previous page...

Register	Offset
TCD30_SOFF	24_8024h
TCD31_SOFF	24_C024h

Function

This register contains the sign-extended value added to Source Address register after each read transaction.

Diagram



Fields

Field	Function
15-0 SOFF	Source Address Signed Offset Sign-extended offset applied to the current source address to form the next-state value as each source read is completed.

14.6.2.9 TCD Transfer Attributes (TCD0_ATTR - TCD31_ATTR)

Offset

Register	Offset
TCD0_ATTR	26h
TCD1_ATTR	4026h
TCD2_ATTR	8026h
TCD3_ATTR	C026h
TCD4_ATTR	1_0026h
TCD5_ATTR	1_4026h
TCD6_ATTR	1_8026h
TCD7_ATTR	1_C026h
TCD8_ATTR	2_0026h
TCD9_ATTR	2_4026h
TCD10_ATTR	2_8026h

Table continues on the next page...

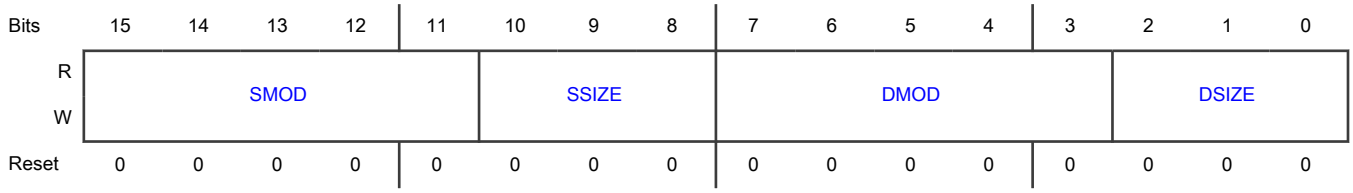
Table continued from the previous page...

Register	Offset
TCD11_ATTR	2_C026h
TCD12_ATTR	20_0026h
TCD13_ATTR	20_4026h
TCD14_ATTR	20_8026h
TCD15_ATTR	20_C026h
TCD16_ATTR	21_0026h
TCD17_ATTR	21_4026h
TCD18_ATTR	21_8026h
TCD19_ATTR	21_C026h
TCD20_ATTR	22_0026h
TCD21_ATTR	22_4026h
TCD22_ATTR	22_8026h
TCD23_ATTR	22_C026h
TCD24_ATTR	23_0026h
TCD25_ATTR	23_4026h
TCD26_ATTR	23_8026h
TCD27_ATTR	23_C026h
TCD28_ATTR	24_0026h
TCD29_ATTR	24_4026h
TCD30_ATTR	24_8026h
TCD31_ATTR	24_C026h

Function

This register contains size and option modulo addressing information for source and destination addresses.

Diagram



Fields

Field	Function
15-11 SMOD	Source Address Modulo This field defines a specific address range, which is the value after the SADDR + SOFF calculation is performed on the original register value. Setting this field makes it easy to implement a circular data queue. For data queues requiring power-of-2-sized bytes, the queue must start at a 0-modulo-size address and the SMOD field must be set to the appropriate value for the queue, freezing the required number of upper address bits. The value programmed into this field specifies the number of lower address bits that are allowed to change. For a circular queue application, you typically set TCDn_SOFF[SOFF] to the transfer size to implement post-increment addressing, with the SMOD function constraining the addresses to a 0-modulo-size range. 0_0000b - Source address modulo feature disabled 0_0001b - Source address modulo feature enabled for any non-zero value [1-31]
10-8 SSIZE	Source Data Transfer Size 000b - 8-bit 001b - 16-bit 010b - 32-bit 011b - 64-bit 100b - 16-byte 101b - 32-byte 110b - 64-byte 111b - Reserved
7-3 DMOD	Destination Address Modulo See the SMOD definition.
2-0 DSIZE	Destination Data Transfer Size See the SSIZE definition.

14.6.2.10 TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO - TCD31_NBYTES_MLOFFNO)

Offset

Register	Offset
TCD0_NBYTES_MLOFFNO	28h
TCD1_NBYTES_MLOFFNO	4028h
TCD2_NBYTES_MLOFFNO	8028h
TCD3_NBYTES_MLOFFNO	C028h
TCD4_NBYTES_MLOFFNO	1_0028h
TCD5_NBYTES_MLOFFNO	1_4028h
TCD6_NBYTES_MLOFFNO	1_8028h
TCD7_NBYTES_MLOFFNO	1_C028h
TCD8_NBYTES_MLOFFNO	2_0028h
TCD9_NBYTES_MLOFFNO	2_4028h
TCD10_NBYTES_MLOFFNO	2_8028h
TCD11_NBYTES_MLOFFNO	2_C028h
TCD12_NBYTES_MLOFFNO	20_0028h
TCD13_NBYTES_MLOFFNO	20_4028h
TCD14_NBYTES_MLOFFNO	20_8028h
TCD15_NBYTES_MLOFFNO	20_C028h
TCD16_NBYTES_MLOFFNO	21_0028h
TCD17_NBYTES_MLOFFNO	21_4028h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD18_NBYTES_MLOFFNO	21_8028h
TCD19_NBYTES_MLOFFNO	21_C028h
TCD20_NBYTES_MLOFFNO	22_0028h
TCD21_NBYTES_MLOFFNO	22_4028h
TCD22_NBYTES_MLOFFNO	22_8028h
TCD23_NBYTES_MLOFFNO	22_C028h
TCD24_NBYTES_MLOFFNO	23_0028h
TCD25_NBYTES_MLOFFNO	23_4028h
TCD26_NBYTES_MLOFFNO	23_8028h
TCD27_NBYTES_MLOFFNO	23_C028h
TCD28_NBYTES_MLOFFNO	24_0028h
TCD29_NBYTES_MLOFFNO	24_4028h
TCD30_NBYTES_MLOFFNO	24_8028h
TCD31_NBYTES_MLOFFNO	24_C028h

Function

The TCDn_NBYTES field defines the number of bytes to transfer per service request.

Minor loop offsets are address offset values added to the final source address (TCDn_SADDR), or destination address (TCDn_DADDR), upon minor loop completion. Minor loop completion is when the channel has finished the service request and has transferred NBYTES. When minor loop offsets are enabled, the minor loop offset value (TCDn_NBYTES_MLOFFYES[MLOFF]) is added to the final source address (TCDn_SADDR), to the final destination address (TCDn_DADDR), or to both, prior to the addresses being written back to the TCD. If the major loop is complete, the minor loop offset is ignored and the major loop address offsets (TCDn_SLAST_SDA and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When minor loop mapping is enabled (SMLOE or DMLOE is 1), TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is redefined. A portion of TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is used to specify multiple fields:

- A source enable bit (SMLOE) to specify the minor loop offset must be applied to the source address (TCDn_SADDR) upon minor loop completion
- A destination enable bit (DMLOE) to specify the minor loop offset must be applied to the destination address (TCDn_DADDR) upon minor loop completion
- The sign extended minor loop offset value (MLOFF)

The same offset value (MLOFF) is used for both source and destination minor loop offsets. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. If both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

One of two register profiles (this register or TCDn_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether source or destination minor loop mapping is enabled.

TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is defined as follows:

- If SMLOE = 0 and DMLOE = 0, then see the TCDn_NBYTES_MLOFFNO register description.
- If either SMLOE or DMLOE is 1, then see the TCDn_NBYTES_MLOFFYES register description.

Diagram

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

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

Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - Minor loop offset not applied to SADDR 1b - Minor loop offset applied to SADDR
30 DMLOE	Destination Minor Loop Offset Enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - Minor loop offset not applied to DADDR 1b - Minor loop offset applied to DADDR
29-0 NBYTES	Number of Bytes To Transfer Per Service Request Number of bytes to be transferred for each service request of the channel.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When a channel activates, the module loads the appropriate TCD contents into the eDMA engine and performs the appropriate reads and writes until the byte transfer count has been reached. This process is normally an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the byte count is exhausted, the SADDR and DADDR values are written back into the TCD memory, and the major loop iteration count (CITER) is decremented by one and written back to the TCD memory. If the major iteration count is complete, additional processing is performed.

14.6.2.11 TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES - TCD31_NBYTES_MLOFFYES)

Offset

Register	Offset
TCD0_NBYTES_MLOFFYES	28h
TCD1_NBYTES_MLOFFYES	4028h
TCD2_NBYTES_MLOFFYES	8028h
TCD3_NBYTES_MLOFFYES	C028h
TCD4_NBYTES_MLOFFYES	1_0028h
TCD5_NBYTES_MLOFFYES	1_4028h
TCD6_NBYTES_MLOFFYES	1_8028h
TCD7_NBYTES_MLOFFYES	1_C028h
TCD8_NBYTES_MLOFFYES	2_0028h
TCD9_NBYTES_MLOFFYES	2_4028h
TCD10_NBYTES_MLOFFYES	2_8028h
TCD11_NBYTES_MLOFFYES	2_C028h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD12_NBYTES_MLOFFYES	20_0028h
TCD13_NBYTES_MLOFFYES	20_4028h
TCD14_NBYTES_MLOFFYES	20_8028h
TCD15_NBYTES_MLOFFYES	20_C028h
TCD16_NBYTES_MLOFFYES	21_0028h
TCD17_NBYTES_MLOFFYES	21_4028h
TCD18_NBYTES_MLOFFYES	21_8028h
TCD19_NBYTES_MLOFFYES	21_C028h
TCD20_NBYTES_MLOFFYES	22_0028h
TCD21_NBYTES_MLOFFYES	22_4028h
TCD22_NBYTES_MLOFFYES	22_8028h
TCD23_NBYTES_MLOFFYES	22_C028h
TCD24_NBYTES_MLOFFYES	23_0028h
TCD25_NBYTES_MLOFFYES	23_4028h
TCD26_NBYTES_MLOFFYES	23_8028h
TCD27_NBYTES_MLOFFYES	23_C028h
TCD28_NBYTES_MLOFFYES	24_0028h
TCD29_NBYTES_MLOFFYES	24_4028h
TCD30_NBYTES_MLOFFYES	24_8028h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD31_NBYTES_MLOFFYES	24_C028h

Function

The TCDn_NBYTES field defines the number of bytes to transfer per service request.

Minor loop offset is an address offset value added to the final source address (TCDn_SADDR) or destination address (TCDn_DADDR) upon minor loop completion. Minor loop completion occurs when the channel has finished the service request and has transferred NBYTES. Minor loop offsets are enabled by setting either the source enable bit (SMLOE) or the destination enable bit (DMLOE).

The source enable bit (SMLOE) specifies the minor loop offset value (MLOFF) that is to be applied to the source address (TCDn_SADDR) upon minor loop completion. The destination enable bit (DMLOE) specifies the minor loop offset (MLOFF) that is to be applied to the destination address (TCDn_DADDR) upon minor loop completion.

If the major loop is complete, the minor loop offsets are ignored and the major loop address offsets (TCDn_SLAST_SDA and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When you enable the minor loop offset overlay (either SMLOE or DMLOE is 1), eDMA redefines TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES. A portion of TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES specifies the sign-extended minor loop offset value (MLOFF). The same offset value (MLOFF) applies to both source and destination minor loop offsets. When the minor loop offset is enabled, you must align it to the transfer size of the source or destination it is associated with. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. If both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

One of two register profiles (this register or TCDn_NBYTES_MLOFFNO) defines the number of bytes to transfer per request. Which register to use depends on whether source or destination minor loop mapping is enabled.

TCDn_NBYTES_MLOFFYES is defined as follows:

- If either minor loop offset is enabled (SMLOE or DMLOE = 1), then see the TCDn_NBYTES_MLOFFYES register description.
- If SMLOE and DMLOE are both 0, then see the TCDn_NBYTES_MLOFFNO register description.

Diagram

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

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

Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - Minor loop offset not applied to SADDR 1b - Minor loop offset applied to SADDR
30 DMLOE	Destination Minor Loop Offset Enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - Minor loop offset not applied to DADDR 1b - Minor loop offset applied to DADDR
29-10 MLOFF	Minor Loop Offset If SMLOE or DMLOE is 1, this field represents a sign-extended offset applied to the source or destination address to form the next-state value after the minor loop completes.
9-0 NBYTES	Number of Bytes To Transfer Per Service Request The number of bytes to be transferred in each service request of the channel. As a channel activates, the module loads the appropriate TCD contents into the eDMA engine and performs the appropriate reads and writes until the minor byte transfer count has been reached. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, and the major iteration count is decremented and restored to the TCD memory. If the major iteration count is complete, additional processing is performed.

14.6.2.12 TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA - TCD31_SLAST_SDA)

Offset

Register	Offset
TCD0_SLAST_SDA	2Ch
TCD1_SLAST_SDA	402Ch
TCD2_SLAST_SDA	802Ch
TCD3_SLAST_SDA	C02Ch
TCD4_SLAST_SDA	1_002Ch
TCD5_SLAST_SDA	1_402Ch
TCD6_SLAST_SDA	1_802Ch
TCD7_SLAST_SDA	1_C02Ch

Table continues on the next page...

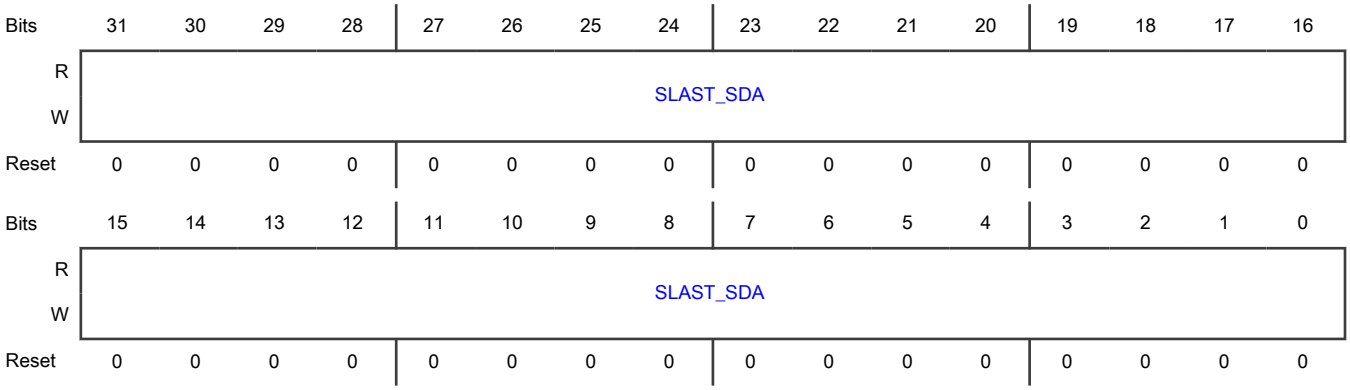
Table continued from the previous page...

Register	Offset
TCD8_SLAST_SDA	2_002Ch
TCD9_SLAST_SDA	2_402Ch
TCD10_SLAST_SDA	2_802Ch
TCD11_SLAST_SDA	2_C02Ch
TCD12_SLAST_SDA	20_002Ch
TCD13_SLAST_SDA	20_402Ch
TCD14_SLAST_SDA	20_802Ch
TCD15_SLAST_SDA	20_C02Ch
TCD16_SLAST_SDA	21_002Ch
TCD17_SLAST_SDA	21_402Ch
TCD18_SLAST_SDA	21_802Ch
TCD19_SLAST_SDA	21_C02Ch
TCD20_SLAST_SDA	22_002Ch
TCD21_SLAST_SDA	22_402Ch
TCD22_SLAST_SDA	22_802Ch
TCD23_SLAST_SDA	22_C02Ch
TCD24_SLAST_SDA	23_002Ch
TCD25_SLAST_SDA	23_402Ch
TCD26_SLAST_SDA	23_802Ch
TCD27_SLAST_SDA	23_C02Ch
TCD28_SLAST_SDA	24_002Ch
TCD29_SLAST_SDA	24_402Ch
TCD30_SLAST_SDA	24_802Ch
TCD31_SLAST_SDA	24_C02Ch

Function

This register contains the value added to the source address when the major loop is complete. When the store destination address option is enabled, this field provides a pointer to memory for storing the final destination address.

Diagram



Fields

Field	Function
31-0 SLAST_SDA	Last Source Address Adjustment / Store DADDR Address Source last address adjustment or the system memory address for destination address (DADDR) storage. If (TCDn_CSR[ESDA] = 0), then: • Adjustment value is added to the source address at the completion of the major iteration count. This value can be used to restore the source address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final source address adjustment. Otherwise: • This address points to the 32-bit-aligned memory location where the destination address (DADDR) is to be stored in system memory. By saving the final destination address in system memory via the ESDA feature, you are able to compute the size of a variable destination data buffer by simply subtracting the beginning DADDR from the final, saved DADDR. This feature is used together with the scatter/gather operation to prevent the loss of the final DADDR, which is overwritten during the scatter/gather operation. The "Store Destination Address" (SDA) value must be a 32-bit-aligned location because the eDMA forces the lower two address bits of the SLAST_SDA field to zero when ESDA is enabled. The module performs this write operation when the major loop is done; that is, when the major iteration count (CITER) decrements to zero.

14.6.2.13 TCD Destination Address (TCD0_DADDR - TCD31_DADDR)

Offset

Register	Offset
TCD0_DADDR	30h
TCD1_DADDR	4030h
TCD2_DADDR	8030h

Table continues on the next page...

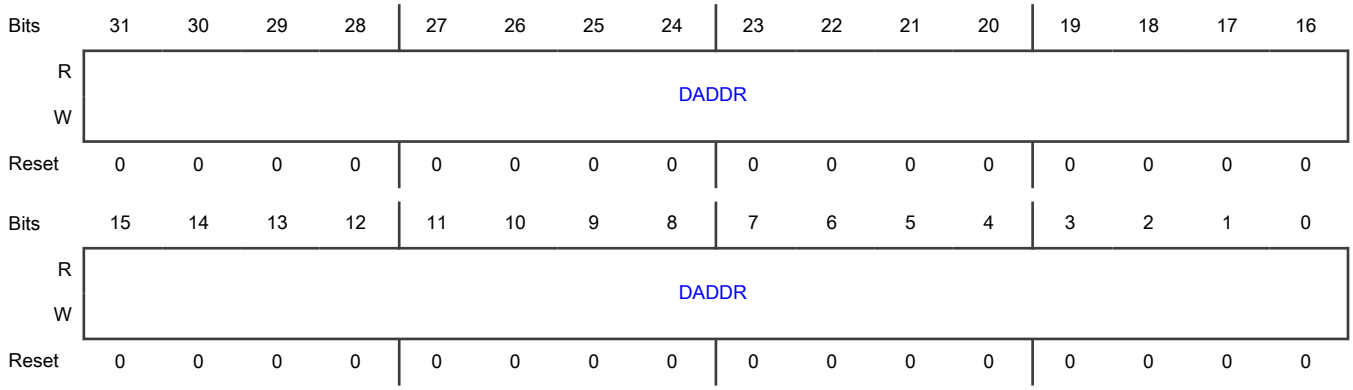
Table continued from the previous page...

Register	Offset
TCD3_DADDR	C030h
TCD4_DADDR	1_0030h
TCD5_DADDR	1_4030h
TCD6_DADDR	1_8030h
TCD7_DADDR	1_C030h
TCD8_DADDR	2_0030h
TCD9_DADDR	2_4030h
TCD10_DADDR	2_8030h
TCD11_DADDR	2_C030h
TCD12_DADDR	20_0030h
TCD13_DADDR	20_4030h
TCD14_DADDR	20_8030h
TCD15_DADDR	20_C030h
TCD16_DADDR	21_0030h
TCD17_DADDR	21_4030h
TCD18_DADDR	21_8030h
TCD19_DADDR	21_C030h
TCD20_DADDR	22_0030h
TCD21_DADDR	22_4030h
TCD22_DADDR	22_8030h
TCD23_DADDR	22_C030h
TCD24_DADDR	23_0030h
TCD25_DADDR	23_4030h
TCD26_DADDR	23_8030h
TCD27_DADDR	23_C030h
TCD28_DADDR	24_0030h
TCD29_DADDR	24_4030h
TCD30_DADDR	24_8030h
TCD31_DADDR	24_C030h

Function

This register contains the address for the write transactions.

Diagram



Fields

Field	Function
31-0 DADDR	Destination Address Memory address pointing to the destination data.

14.6.2.14 TCD Signed Destination Address Offset (TCD0_DOFF - TCD31_DOFF)

Offset

Register	Offset
TCD0_DOFF	34h
TCD1_DOFF	4034h
TCD2_DOFF	8034h
TCD3_DOFF	C034h
TCD4_DOFF	1_0034h
TCD5_DOFF	1_4034h
TCD6_DOFF	1_8034h
TCD7_DOFF	1_C034h
TCD8_DOFF	2_0034h
TCD9_DOFF	2_4034h
TCD10_DOFF	2_8034h
TCD11_DOFF	2_C034h
TCD12_DOFF	20_0034h
TCD13_DOFF	20_4034h
TCD14_DOFF	20_8034h

Table continues on the next page...

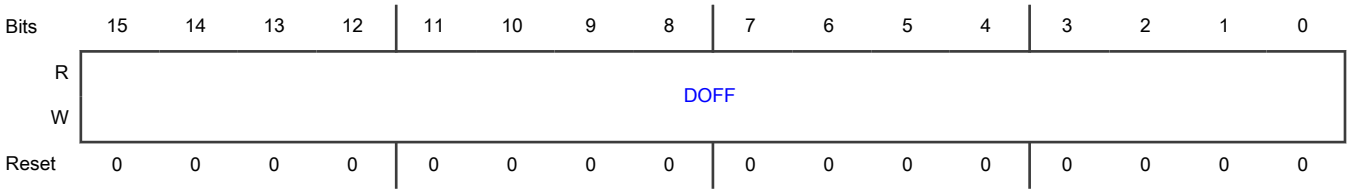
Table continued from the previous page...

Register	Offset
TCD15_DOFF	20_C034h
TCD16_DOFF	21_0034h
TCD17_DOFF	21_4034h
TCD18_DOFF	21_8034h
TCD19_DOFF	21_C034h
TCD20_DOFF	22_0034h
TCD21_DOFF	22_4034h
TCD22_DOFF	22_8034h
TCD23_DOFF	22_C034h
TCD24_DOFF	23_0034h
TCD25_DOFF	23_4034h
TCD26_DOFF	23_8034h
TCD27_DOFF	23_C034h
TCD28_DOFF	24_0034h
TCD29_DOFF	24_4034h
TCD30_DOFF	24_8034h
TCD31_DOFF	24_C034h

Function

This register contains the sign-extended value added to Destination Address register after each write transaction.

Diagram



Fields

Field	Function
15-0 DOFF	Destination Address Signed Offset Sign-extended offset that is applied to the current destination address to form the next-state value as each destination write is completed.

14.6.2.15 TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD31_CITER_ELINKNO)

Offset

Register	Offset
TCD0_CITER_ELINKNO	36h
TCD1_CITER_ELINKNO	4036h
TCD2_CITER_ELINKNO	8036h
TCD3_CITER_ELINKNO	C036h
TCD4_CITER_ELINKNO	1_0036h
TCD5_CITER_ELINKNO	1_4036h
TCD6_CITER_ELINKNO	1_8036h
TCD7_CITER_ELINKNO	1_C036h
TCD8_CITER_ELINKNO	2_0036h
TCD9_CITER_ELINKNO	2_4036h
TCD10_CITER_ELINKNO	2_8036h
TCD11_CITER_ELINKNO	2_C036h
TCD12_CITER_ELINKNO	20_0036h
TCD13_CITER_ELINKNO	20_4036h
TCD14_CITER_ELINKNO	20_8036h
TCD15_CITER_ELINKNO	20_C036h
TCD16_CITER_ELINKNO	21_0036h
TCD17_CITER_ELINKNO	21_4036h
TCD18_CITER_ELINKNO	21_8036h
TCD19_CITER_ELINKNO	21_C036h
TCD20_CITER_ELINKNO	22_0036h
TCD21_CITER_ELINKNO	22_4036h

Table continues on the next page...

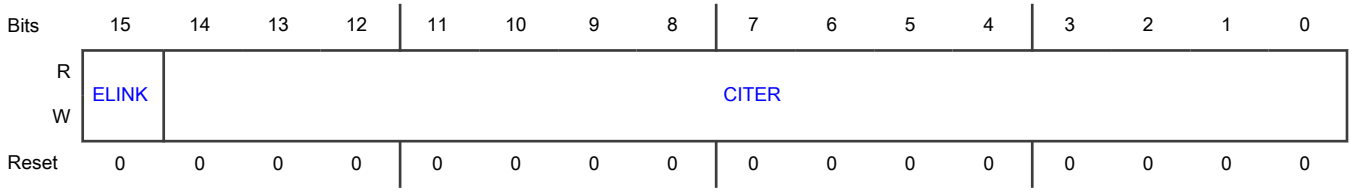
Table continued from the previous page...

Register	Offset
TCD22_CITER_ELINKNO	22_8036h
TCD23_CITER_ELINKNO	22_C036h
TCD24_CITER_ELINKNO	23_0036h
TCD25_CITER_ELINKNO	23_4036h
TCD26_CITER_ELINKNO	23_8036h
TCD27_CITER_ELINKNO	23_C036h
TCD28_CITER_ELINKNO	24_0036h
TCD29_CITER_ELINKNO	24_4036h
TCD30_CITER_ELINKNO	24_8036h
TCD31_CITER_ELINKNO	24_C036h

Function

If TCDn_CITER[ELINK] is 0, the TCDn_CITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by the relevant LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel to 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of MAJORELINK channel linking. **NOTE** This field must be equal to the BITER[ELINK] field; otherwise, a configuration error is reported. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14-0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the channel finishes a service request and is written back to TCD memory. After the major iteration count is exhausted, the channel performs a number of operations — for example, final source and destination address calculations — and optionally generates an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

14.6.2.16 TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD31_CITER_ELINKYES)

Offset

Register	Offset
TCD0_CITER_ELINKYES	36h
TCD1_CITER_ELINKYES	4036h
TCD2_CITER_ELINKYES	8036h
TCD3_CITER_ELINKYES	C036h
TCD4_CITER_ELINKYES	1_0036h
TCD5_CITER_ELINKYES	1_4036h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD6_CITER_ELINKYES	1_8036h
TCD7_CITER_ELINKYES	1_C036h
TCD8_CITER_ELINKYES	2_0036h
TCD9_CITER_ELINKYES	2_4036h
TCD10_CITER_ELINKYES	2_8036h
TCD11_CITER_ELINKYES	2_C036h
TCD12_CITER_ELINKYES	20_0036h
TCD13_CITER_ELINKYES	20_4036h
TCD14_CITER_ELINKYES	20_8036h
TCD15_CITER_ELINKYES	20_C036h
TCD16_CITER_ELINKYES	21_0036h
TCD17_CITER_ELINKYES	21_4036h
TCD18_CITER_ELINKYES	21_8036h
TCD19_CITER_ELINKYES	21_C036h
TCD20_CITER_ELINKYES	22_0036h
TCD21_CITER_ELINKYES	22_4036h
TCD22_CITER_ELINKYES	22_8036h
TCD23_CITER_ELINKYES	22_C036h
TCD24_CITER_ELINKYES	23_0036h

Table continues on the next page...

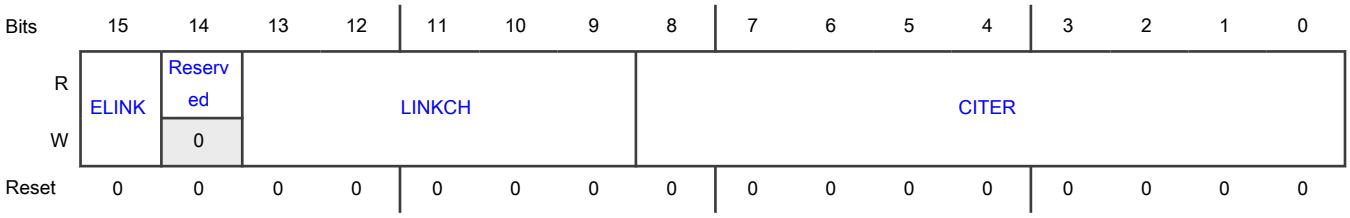
Table continued from the previous page...

Register	Offset
TCD25_CITER_ELINKYES	23_4036h
TCD26_CITER_ELINKYES	23_8036h
TCD27_CITER_ELINKYES	23_C036h
TCD28_CITER_ELINKYES	24_0036h
TCD29_CITER_ELINKYES	24_4036h
TCD30_CITER_ELINKYES	24_8036h
TCD31_CITER_ELINKYES	24_C036h

Function

If TCDn_CITER[ELINK] is 1, the TCDn_CITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by the relevant LINKCH field. When enabled, an internal mechanism sets the TCDn_CSR[START] field of the specified channel (LINKCH) upon minor loop completion. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of MAJORELINK channel linking. **NOTE** This field must be equal to the BITER[ELINK] field; otherwise, a configuration error is reported.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14 —	Reserved
13-9 LINKCH	Minor Loop Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted the eDMA engine initiates a channel service request to the channel defined by this field by writing that channel's TCDn_CSR[START] field to 1.
8-0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the channel finishes a service request and is written back to the TCD memory. After the major iteration count is exhausted, the channel performs a number of operations — for example, final source and destination address calculations — and optionally generates an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

14.6.2.17 TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA - TCD31_DLAST_SGA)

Offset

Register	Offset
TCD0_DLAST_SGA	38h
TCD1_DLAST_SGA	4038h
TCD2_DLAST_SGA	8038h
TCD3_DLAST_SGA	C038h
TCD4_DLAST_SGA	1_0038h
TCD5_DLAST_SGA	1_4038h
TCD6_DLAST_SGA	1_8038h
TCD7_DLAST_SGA	1_C038h

Table continues on the next page...

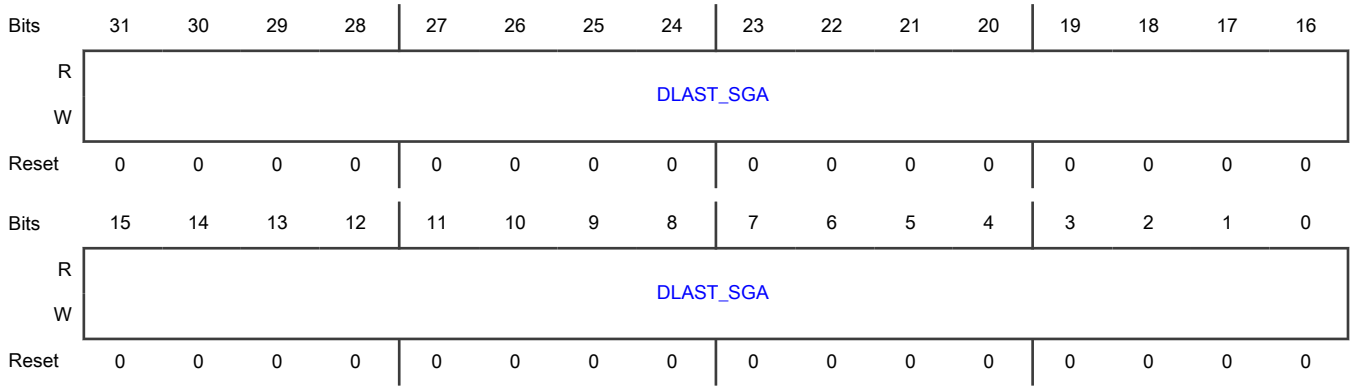
Table continued from the previous page...

Register	Offset
TCD8_DLAST_SGA	2_0038h
TCD9_DLAST_SGA	2_4038h
TCD10_DLAST_SGA	2_8038h
TCD11_DLAST_SGA	2_C038h
TCD12_DLAST_SGA	20_0038h
TCD13_DLAST_SGA	20_4038h
TCD14_DLAST_SGA	20_8038h
TCD15_DLAST_SGA	20_C038h
TCD16_DLAST_SGA	21_0038h
TCD17_DLAST_SGA	21_4038h
TCD18_DLAST_SGA	21_8038h
TCD19_DLAST_SGA	21_C038h
TCD20_DLAST_SGA	22_0038h
TCD21_DLAST_SGA	22_4038h
TCD22_DLAST_SGA	22_8038h
TCD23_DLAST_SGA	22_C038h
TCD24_DLAST_SGA	23_0038h
TCD25_DLAST_SGA	23_4038h
TCD26_DLAST_SGA	23_8038h
TCD27_DLAST_SGA	23_C038h
TCD28_DLAST_SGA	24_0038h
TCD29_DLAST_SGA	24_4038h
TCD30_DLAST_SGA	24_8038h
TCD31_DLAST_SGA	24_C038h

Function

This register contains the value added to the destination address when the major loop is complete. When the Scatter/Gather option is enabled, this field provides a pointer to memory for fetching a transfer control descriptor to reprogram the channel.

Diagram



Fields

Field	Function
31-0 DLAST_SGA	Last Destination Address Adjustment / Scatter Gather Address Adjustment of the last destination address or the memory address for the next transfer control descriptor to be loaded into this channel (scatter/gather). If (TCDn_CSR[ESG] = 0) then: • Adjustment value is added to the destination address at the completion of the major iteration count. This value can apply to restore the destination address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final destination address adjustment. Otherwise: • This address points to the beginning of a 0-modulo 32-byte region containing the next transfer control descriptor to be loaded into this channel. This channel reload is performed as the major iteration count completes. The scatter/gather address must be 0-modulo 32-byte, or else a configuration error is reported.

14.6.2.18 TCD Control and Status (TCD0_CSR - TCD31_CSR)

Offset

Register	Offset
TCD0_CSR	3Ch
TCD1_CSR	403Ch
TCD2_CSR	803Ch
TCD3_CSR	C03Ch
TCD4_CSR	1_003Ch
TCD5_CSR	1_403Ch
TCD6_CSR	1_803Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD7_CSR	1_C03Ch
TCD8_CSR	2_003Ch
TCD9_CSR	2_403Ch
TCD10_CSR	2_803Ch
TCD11_CSR	2_C03Ch
TCD12_CSR	20_003Ch
TCD13_CSR	20_403Ch
TCD14_CSR	20_803Ch
TCD15_CSR	20_C03Ch
TCD16_CSR	21_003Ch
TCD17_CSR	21_403Ch
TCD18_CSR	21_803Ch
TCD19_CSR	21_C03Ch
TCD20_CSR	22_003Ch
TCD21_CSR	22_403Ch
TCD22_CSR	22_803Ch
TCD23_CSR	22_C03Ch
TCD24_CSR	23_003Ch
TCD25_CSR	23_403Ch
TCD26_CSR	23_803Ch
TCD27_CSR	23_C03Ch
TCD28_CSR	24_003Ch
TCD29_CSR	24_403Ch
TCD30_CSR	24_803Ch
TCD31_CSR	24_C03Ch

Function

This register is used to enable optional features.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BWC			MAJORLINKCH					ESDA	EEOP	MAJOREL...	ESG	DREQ	INTHALF	INTMAJOR	START
W			0													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 BWC	Bandwidth Control Throttles the amount of bus bandwidth consumed by the eDMA. Generally, as the eDMA processes the minor loop, it continuously generates read/write sequences until the minor count is exhausted. This field forces eDMA to stall after the completion of each read/write access, to control the bus request bandwidth seen by the system bus interconnect. **NOTE** If the source and destination sizes are equal, this field is ignored between the first and second transfers and after the last write of each minor loop. This behavior is a side effect of reducing start-up latency. 00b - No eDMA engine stalls 01b - Enable eDMA master high-priority elevation (HPE) mode. No eDMA engine stalls. 10b - eDMA engine stalls for 4 cycles after each R/W 11b - eDMA engine stalls for 8 cycles after each R/W
13 —	Reserved
12-8 MAJORLINKCH	Major Loop Link Channel Number If (MAJORELINK = 0) then: • No channel-to-channel linking, or chaining, is performed after the major loop counter is exhausted. Otherwise: • After the major loop counter is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] field to 1.
7 ESDA	Enable Store Destination Address As the channel completes the major loop by either the current iteration counter (CITER) decrementing to 0, or by receiving an enabled end-of-packet signal, this field enables writing the destination address (DADDR) to the address stored in the SLAST_SDA field. The value written to system memory is the next DADDR value prior to the DLAST_SGA offset being applied, or overwritten by an enabled scatter/gather operation. When the ESDA bit is 1, SLAST_SDA contains the write pointer instead of the final source address offset. Because this is a pointer and not a final offset, a last source address offset of zero is applied to SADDR instead of the SLAST_SGA value. 0b - Ability to store destination address to system memory disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Ability to store destination address to system memory enabled
6 EEOP	Enable End-Of-Packet Processing When enabled by the EEOP field, an end-of-packet hardware input signal directs eDMA to discontinue executing the active channel, and to treat the shutdown as the major-loop-completed event. If the EEOP field is 1, the end-of-packet signal from supported peripherals is accepted. If the EEOP field is 0, the end-of-packet input is ignored. With an end-of-packet retirement, the current TCD destination address (or ESDA-saved destination address), minus the software-saved initial address (DADDR), reflects the total amount of data transferred. 0b - End-of-packet operation disabled 1b - End-of-packet hardware input signal enabled
5 MAJORELINK	Enable Link When Major Loop Complete As the channel completes the major loop, this flag enables linking to another channel defined by MAJORLINKCH. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. **NOTE** To support the dynamic linking coherency model, this field is forced to 0 if written when TCDn_CSR[DONE] is 1. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
4 ESG	Enable Scatter/Gather Processing As the channel completes the major loop, this flag enables scatter/gather processing in the current channel. If enabled, the eDMA engine uses TCDn_DLAST_SGA as a memory pointer to a 0-modulo 32-bit address containing a 32-byte data structure, which is loaded as the transfer control descriptor into local memory. **NOTE** To support the dynamic scatter/gather coherency model, this field is forced to 0 if written when TCDn_CSR[DONE] is 1. 0b - Current channel's TCD is normal format 1b - Current channel's TCD specifies scatter/gather format.
3 DREQ	Disable Request If this flag is 1, the eDMA hardware automatically clears the corresponding ERQ bit when the current major iteration count reaches 0. 0b - No operation. Channel's ERQ field not affected 1b - Clear the ERQ field to 0 upon major loop completion, thus disabling hardware service requests. Channel's ERQ field cleared to 0 when major loop complete
2	Enable Interrupt If Major Counter Half-complete

Table continues on the next page...

Table continued from the previous page...

Field	Function
INTHALF	If this flag is 1, the channel generates an interrupt request by setting the appropriate field in the INT register to 1 when the current major iteration count reaches the halfway point. Specifically, the comparison performed by the eDMA engine is (CITER = (BITER/2)). This halfway point interrupt request is provided to support double-buffered, also known as ping-pong, schemes, or other types of data movement where the processor needs an early indication of the transfer's progress. **NOTE** If BITER = 1, do not use INTHALF; use INTMAJOR instead. 0b - Halfway point interrupt disabled 1b - Halfway point interrupt enabled
1 INTMAJOR	Enable Interrupt If Major count complete If this flag is 1, the channel generates an interrupt request by setting the appropriate field in the INT register to 1 when the current major iteration count (CITER) reaches 0. 0b - End-of-major loop interrupt disabled 1b - End-of-major loop interrupt enabled
0 START	Channel Start If this flag is 1, the channel is requesting service. The eDMA hardware automatically clears this flag to 0 after the channel begins execution. 0b - Channel not explicitly started 1b - Channel explicitly started via a software-initiated service request

14.6.2.19 TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD31_BITER_ELINKNO)

Offset

Register	Offset
TCD0_BITER_ELINKNO	3Eh
TCD1_BITER_ELINKNO	403Eh
TCD2_BITER_ELINKNO	803Eh
TCD3_BITER_ELINKNO	C03Eh
TCD4_BITER_ELINKNO	1_003Eh
TCD5_BITER_ELINKNO	1_403Eh
TCD6_BITER_ELINKNO	1_803Eh
TCD7_BITER_ELINKNO	1_C03Eh
TCD8_BITER_ELINKNO	2_003Eh

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD9_BITER_ELINKNO	2_403Eh
TCD10_BITER_ELINKNO	2_803Eh
TCD11_BITER_ELINKNO	2_C03Eh
TCD12_BITER_ELINKNO	20_003Eh
TCD13_BITER_ELINKNO	20_403Eh
TCD14_BITER_ELINKNO	20_803Eh
TCD15_BITER_ELINKNO	20_C03Eh
TCD16_BITER_ELINKNO	21_003Eh
TCD17_BITER_ELINKNO	21_403Eh
TCD18_BITER_ELINKNO	21_803Eh
TCD19_BITER_ELINKNO	21_C03Eh
TCD20_BITER_ELINKNO	22_003Eh
TCD21_BITER_ELINKNO	22_403Eh
TCD22_BITER_ELINKNO	22_803Eh
TCD23_BITER_ELINKNO	22_C03Eh
TCD24_BITER_ELINKNO	23_003Eh
TCD25_BITER_ELINKNO	23_403Eh
TCD26_BITER_ELINKNO	23_803Eh
TCD27_BITER_ELINKNO	23_C03Eh

Table continues on the next page...

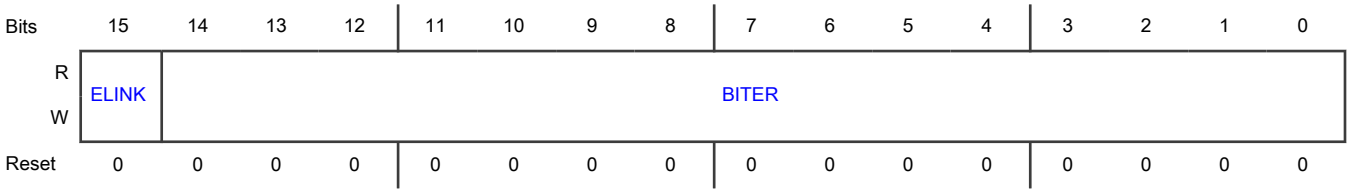
Table continued from the previous page...

Register	Offset
TCD28_BITER_ELINKNO	24_003Eh
TCD29_BITER_ELINKNO	24_403Eh
TCD30_BITER_ELINKNO	24_803Eh
TCD31_BITER_ELINKNO	24_C03Eh

Function

If the TCDn_BITER[ELINK] field is 0, the TCDn_BITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enables Link As the channel completes the minor loop, this flag enables linking to another channel as defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. If channel linking is disabled, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14-0 BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be set equal to the value in the CITER field. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER must be 0x0001.

14.6.2.20 TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD31_BITER_ELINKYES)

Offset

Register	Offset
TCD0_BITER_ELINKYES	3Eh
TCD1_BITER_ELINKYES	403Eh
TCD2_BITER_ELINKYES	803Eh
TCD3_BITER_ELINKYES	C03Eh
TCD4_BITER_ELINKYES	1_003Eh
TCD5_BITER_ELINKYES	1_403Eh
TCD6_BITER_ELINKYES	1_803Eh
TCD7_BITER_ELINKYES	1_C03Eh
TCD8_BITER_ELINKYES	2_003Eh
TCD9_BITER_ELINKYES	2_403Eh
TCD10_BITER_ELINKYES	2_803Eh
TCD11_BITER_ELINKYES	2_C03Eh
TCD12_BITER_ELINKYES	20_003Eh
TCD13_BITER_ELINKYES	20_403Eh
TCD14_BITER_ELINKYES	20_803Eh
TCD15_BITER_ELINKYES	20_C03Eh
TCD16_BITER_ELINKYES	21_003Eh
TCD17_BITER_ELINKYES	21_403Eh

Table continues on the next page...

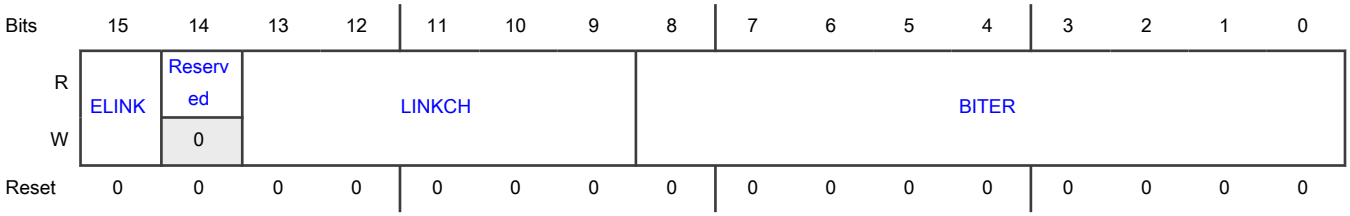
Table continued from the previous page...

Register	Offset
TCD18_BITER_ELINKYES	21_803Eh
TCD19_BITER_ELINKYES	21_C03Eh
TCD20_BITER_ELINKYES	22_003Eh
TCD21_BITER_ELINKYES	22_403Eh
TCD22_BITER_ELINKYES	22_803Eh
TCD23_BITER_ELINKYES	22_C03Eh
TCD24_BITER_ELINKYES	23_003Eh
TCD25_BITER_ELINKYES	23_403Eh
TCD26_BITER_ELINKYES	23_803Eh
TCD27_BITER_ELINKYES	23_C03Eh
TCD28_BITER_ELINKYES	24_003Eh
TCD29_BITER_ELINKYES	24_403Eh
TCD30_BITER_ELINKYES	24_803Eh
TCD31_BITER_ELINKYES	24_C03Eh

Function

If the TCDn_BITER[ELINK] field is set, the TCDn_BITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. If channel linking disables, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14 —	Reserved
13-9 LINKCH	Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] field. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field.
8-0 BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be set equal to the value in the CITER field. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER must be 0x0001.

14.7 Glossary

TCD Transfer control descriptor

Chapter 15
Interrupt Monitor (INTM)

15.1 Overview

INTM provides a mechanism to monitor the latency of the responses on interrupt requests to ensure that the processing of these critical interrupts executes within the expected time frame, increasing the reliability of the device.

15.1.1 Block diagram

The following block diagram shows the major registers involved in monitoring the interrupt sources.

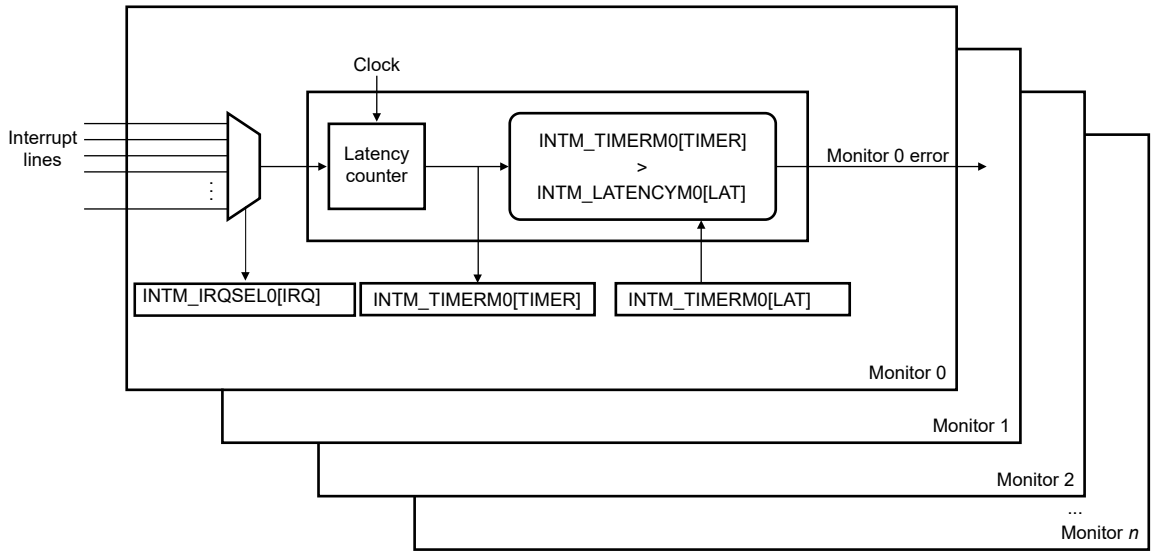


Figure 37. INTM block diagram

15.1.2 Features

INTM supports the following features:

- 4 programmable interrupt monitors
- Programmable interrupt source per monitor
- Programmable 24-bit maximum latency counter per monitor
- Timer expired status bit per monitor
- One interrupt acknowledge for all monitors

15.2 Functional description

Interrupt Request Select for Monitor a (INTM_IRQSEL0 - INTM_IRQSEL3) provides selection of the source, Monitor Mode (INTM_MM) enables the monitor, Interrupt Acknowledge (INTM_IACK) captures the event when the interrupt is acknowledged, Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3) can be programmed to trigger a monitor error when a threshold value is exceeded, Status for Monitor a (INTM_STATUS0 - INTM_STATUS3) is available to read the monitor error over peripheral bus.

NOTE

INTM supports only level-type interrupts. Hence, the pulse-type interrupts are converted into level-type interrupts for INTM.

15.2.1 Clocking

This module has no clocking considerations.

15.2.2 Interrupts

This module has no interrupts.

15.3 External signals

This module has no external signals.

15.4 Initialization

To monitor a subset of interrupt sources:

1. Program Interrupt Request Select for Monitor a (INTM_IRQSEL0 - INTM_IRQSEL3) with a value corresponding to the interrupt source number to be monitored.
   - You can monitor only defined interrupt sources.
2. Program Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3) to the maximum expected latency time for each monitored interrupt source.
3. Write 1 to INTM_MM[MM] to monitor the registers.
4. During the interrupt service routine, write the IRQ of the ISR to Interrupt Acknowledge (INTM_IACK).
5. If the maximum expected latency time exceeds the limit defined in Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3), then an error indication is sent to Fault Collection Control Unit(FCCU). You can also read Status for Monitor a (INTM_STATUS0 - INTM_STATUS3) to see this information.

To clear an error condition, write 0 to the corresponding INTM_TIMERa[TIMER] field for which the interrupt source exceeded the programmed latency value. In case the source is unknown, NXP recommends you to write 0 to all INTM_TIMERa[TIMER] fields.

15.5 INTM register descriptions

INTM allows you to set a maximum timer count for the interrupt latency from interrupt request to interrupt acknowledge. This mechanism monitors the latency of interrupt sources to ensure that these critical interrupts execute within the expected time frame, increasing the reliability of the chip. The hardware for this function is isolated in its own hierarchy to decrease the risk of fault interference.

An error indication is sent to FCCU if the interrupt processing latency exceeds the programmed threshold of the expected latency. The error indication can be read from INTM_STATUSa[STATUS].

15.5.1 INTM memory map

INTM base address: 4027_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Monitor Mode (INTM_MM)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4h	Interrupt Acknowledge (INTM_IACK)	32	RW	0000_0000h
8h	Interrupt Request Select for Monitor 0 (INTM_IRQSEL0)	32	RW	0000_0000h
Ch	Interrupt Latency for Monitor 0 (INTM_LATENCY0)	32	RW	0000_0000h
10h	Timer for Monitor 0 (INTM_TIMER0)	32	RW	0000_0000h
14h	Status for Monitor 0 (INTM_STATUS0)	32	R	0000_0000h
18h	Interrupt Request Select for Monitor 1 (INTM_IRQSEL1)	32	RW	0000_0000h
1Ch	Interrupt Latency for Monitor 1 (INTM_LATENCY1)	32	RW	0000_0000h
20h	Timer for Monitor 1 (INTM_TIMER1)	32	RW	0000_0000h
24h	Status for Monitor 1 (INTM_STATUS1)	32	R	0000_0000h
28h	Interrupt Request Select for Monitor 2 (INTM_IRQSEL2)	32	RW	0000_0000h
2Ch	Interrupt Latency for Monitor 2 (INTM_LATENCY2)	32	RW	0000_0000h
30h	Timer for Monitor 2 (INTM_TIMER2)	32	RW	0000_0000h
34h	Status for Monitor 2 (INTM_STATUS2)	32	R	0000_0000h
38h	Interrupt Request Select for Monitor 3 (INTM_IRQSEL3)	32	RW	0000_0000h
3Ch	Interrupt Latency for Monitor 3 (INTM_LATENCY3)	32	RW	0000_0000h
40h	Timer for Monitor 3 (INTM_TIMER3)	32	RW	0000_0000h
44h	Status for Monitor 3 (INTM_STATUS3)	32	R	0000_0000h

15.5.2 Monitor Mode (INTM_MM)

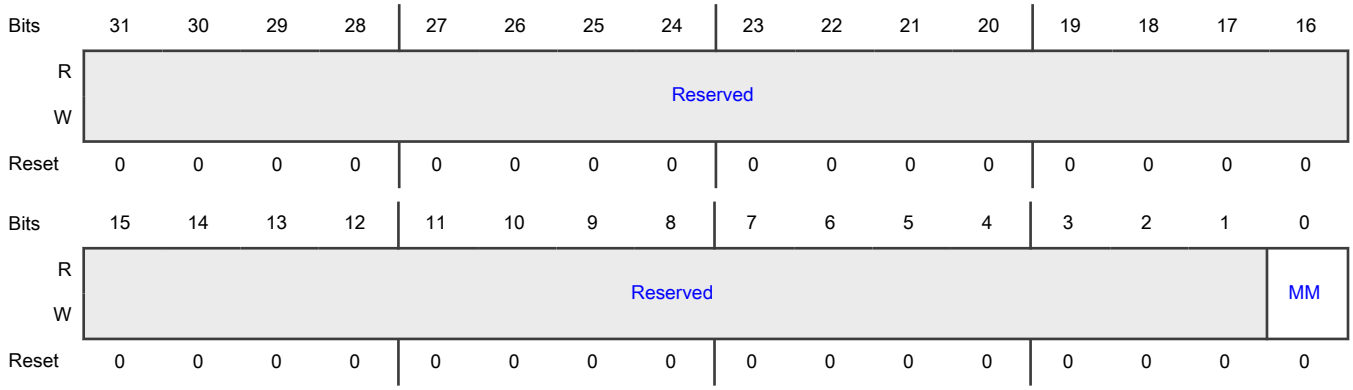
Offset

Register	Offset
INTM_MM	0h

Function

Enables the cycle count timer on a monitored interrupt request for comparison to the latency register.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 MM	Monitor Mode Controls whether the INTM monitors the latency of an interrupt response. 0b - Disable 1b - Enable

15.5.3 Interrupt Acknowledge (INTM_IACK)

Offset

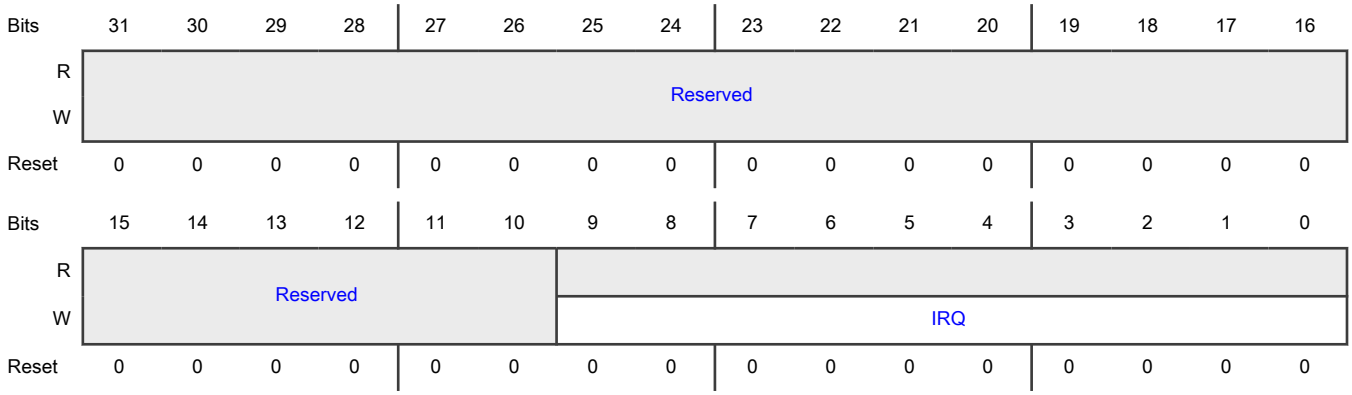
Register	Offset
INTM_IACK	4h

Function

Acknowledges the interrupt processing.

The Interrupt service routine writes to this register after the register acknowledges interrupt processing. This write operation must provide the number of the processed interrupt, which is compared with the interrupt request number in each Interrupt Request Select for Monitor a (INTM_IRQSEL0 - INTM_IRQSEL3) and stops the corresponding Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3) when it found a match.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 IRQ	Interrupt Request Specifies the interrupt request number to stop Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3).

15.5.4 Interrupt Request Select for Monitor a (INTM_IRQSEL0 - INTM_IRQSEL3)

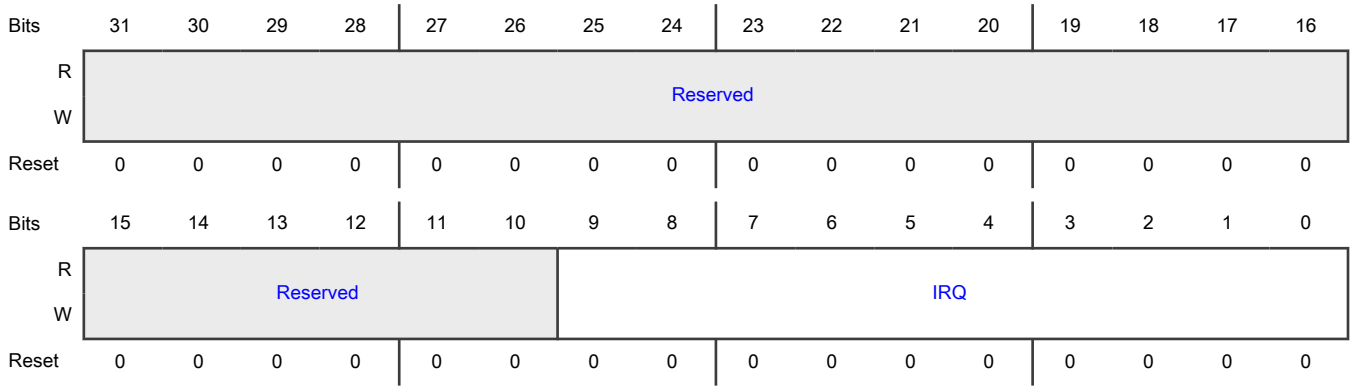
Offset

Register	Offset
INTM_IRQSEL0	8h
INTM_IRQSEL1	18h
INTM_IRQSEL2	28h
INTM_IRQSEL3	38h

Function

Indicates which interrupt request must be monitored or checked.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 IRQ	Interrupt Request Selects the interrupt request number to monitor.

15.5.5 Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3)

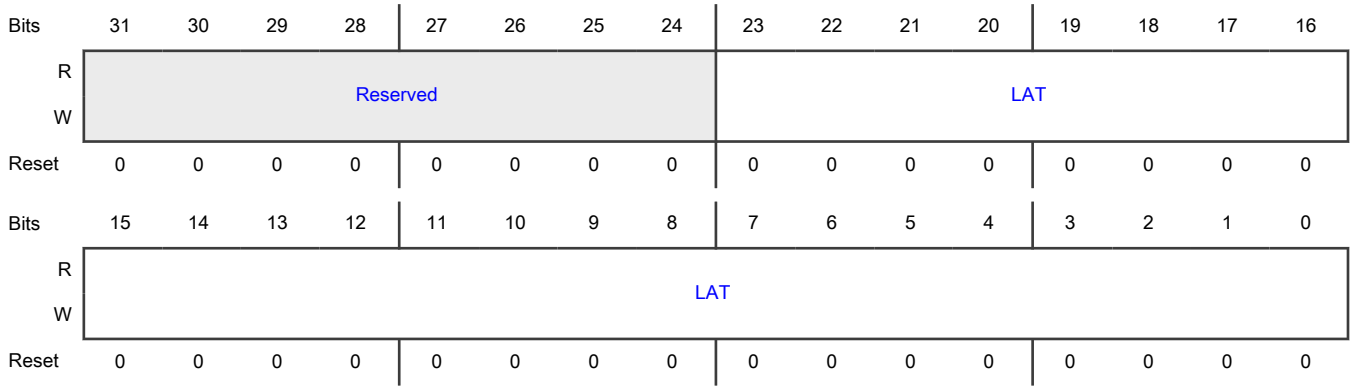
Offset

Register	Offset
INTM_LATENCY0	Ch
INTM_LATENCY1	1Ch
INTM_LATENCY2	2Ch
INTM_LATENCY3	3Ch

Function

Indicates the maximum time before an error signal is asserted for a detected interrupt request to interrupt acknowledge.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 LAT	Latency Specifies the maximum number of INTM clock cycles allowed for the monitored interrupt request. Maximum programmed value must not exceed FF_FFFDh to allow for proper timer error capture.

15.5.6 Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3)

Offset

Register	Offset
INTM_TIMER0	10h
INTM_TIMER1	20h
INTM_TIMER2	30h
INTM_TIMER3	40h

Function

Counts the number of INTM clock cycles for a detected interrupt request to an acknowledged interrupt processing.

Initializes to 1 in following conditions:

- Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3) is nonzero
- Monitor error is not asserted
- Zero to one transition on the monitored interrupt request

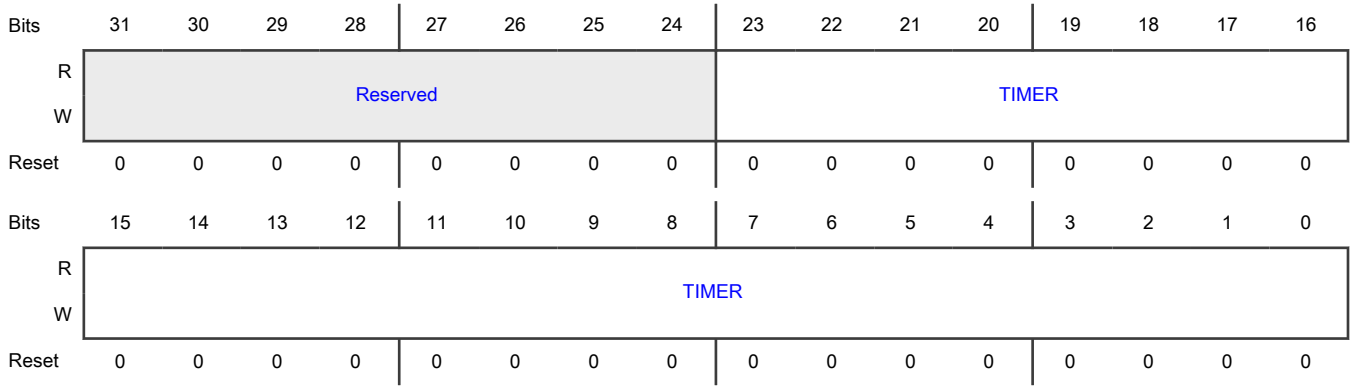
Following are the conditions to enable Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3):

- INTM_MM = 1 and zero-to-one transition on the monitored interrupt source

Following are the conditions to disable Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3):

- Reset
- INTM_MM = 1 and monitor error
- INTM_MM = 1 and monitored interrupt request is not asserted
- Interrupt acknowledge for corresponding interrupt request
- After the timer is enabled, writing 0 to INTM_MM[MM] has no effect.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 TIMER	Timer Counts the number of INTM clock cycles up to 24 bits of resolution.

15.5.7 Status for Monitor a (INTM_STATUS0 - INTM_STATUS3)

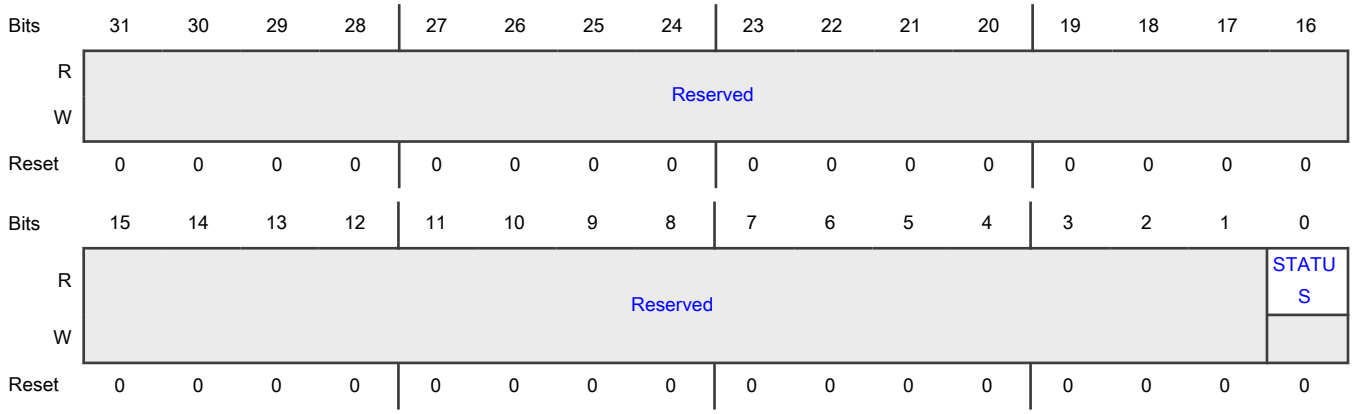
Offset

Register	Offset
INTM_STATUS0	14h
INTM_STATUS1	24h
INTM_STATUS2	34h
INTM_STATUS3	44h

Function

Defines the monitor status. Indicates whether Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3) value has exceeded Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3) value. You can clear the monitor status by writing 0 to the corresponding INTM_TIMERa[TIMER].

Diagram



Fields

Field	Function
31-1 —	Reserved
0 STATUS	Monitor status Indicates whether Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3) value has exceeded the Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3) value. 0b - Did not exceed 1b - Exceeded

Chapter 16
Semaphores2 (SEMA42)

16.1 Chip-specific SEMA42 information

16.1.1 SEMA42 instance and configuration

This chip has up to one instance of SEMA42. See the following table for availability of SEMA42 instance across MWCT2xxxS product series.

Table 51. SEMA42 instance

Instance	MWCT2D17S/MWCT2D16S	MWCT2014S/ MWCT2015S/MWCT2016S
SEMA42[1]	Yes	No

1. Load- and store-exclusive instructions are supported at the core level, but on multi-core devices the SEMA42 must be used.

See "Chip-specific XRDC information" for the domain ID of each master on the MWCT2xxxS product series. The following table shows the number of domains per chip:

Table 52. Number of domains

Chip	Number of domains
MWCT2014S/MWCT2015S/MWCT2016S	Not available
MWCT2D16S	3
MWCT2D17S	3

16.2 Introduction

SEMA42 is a memory-mapped module that provides robust hardware support needed in multi-core systems for implementing semaphores and provides a simple mechanism to achieve "lock and unlock" operations via a single - write access. The hardware semaphore module provides hardware-enforced gates as well as other useful system functions related to the gating mechanisms.

16.2.1 Multi-core programming: software gates

Multi-processor systems require a function that can be used to safely and easily provide a locking mechanism for system software to control access to shared data structures, shared hardware resources, and so on. The software uses the gating mechanisms to serialize (and synchronize) accesses to shared data and/or resources to prevent race conditions and preserve memory coherency between different processes and domains.

Consider the following description of a typical use-case: Domain X enters a section of code where shared data values are to be updated. The domain must first acquire a semaphore. Think of this as the locking (or closing) of a software gate. After the gate locks, a properly-architected software system does not allow other processes (or domains) to execute the same code segment or modify the shared data structure protected by the gate. In other words, the system locks out other processes/domains. Many software implementations include a spin-wait loop within the lock function until the gate locks. After domain X obtains the lock, domain X continues execution and updates the data values protected by the particular lock. After domain X completes the updates, it unlocks (or opens) the software gate, allowing other processes/domains access to the updated data values.

A correctly-implemented system solution must follow these important rules:

- A gate variable must protect all writes to shared data values or shared hardware resources.
- After a domain locks a gate, the system must block other processes/domains from accessing the shared data or resources. Software conventions enforce this.

- The domain that locks a particular gate is the only domain that can open (unlock) that gate.

Information in the hardware gate identifying the locking domain can be extremely useful for system-level debugging.

16.2.2 Features

SEMA42 implements hardware-enforced semaphores as an IPS-mapped slave peripheral device. The feature set includes:

- Support for 16 hardware-enforced gates in a multi-domain configuration that supports up to 15 domains. In this chapter, dm*X* represents domain *X*.
  - Gates appear as a 16-entry byte-size array with read and write accesses.
    - Domains lock gates by writing "domainID_number+1" to the appropriate gate and must read back the gate value to verify that the lock operation succeeded.
    - After the gate locks, the locking domain unlocks the gate by writing zeroes.
  - Each hardware gate appears as a 16-state, 4-bit state machine.
    - 16-state implementation

      If gate = 0h, then state = unlocked

      if gate = 1h, then state = locked by domain (master) 0

      if gate = 2h, then state = locked by domain (master) 1

      …

      if gate = Fh, then state = locked by domain (master) 14
    - SEMA42 uses the logical domain number and the specified data patterns as reference attributes to validate all write operation.
    - After a gate locks, the locking domain must unlock the gate by writing zeroes.
  - Support for secure reset mechanisms to clear the contents of individual gates, as well as a clear-all capability
- Memory-mapped slave peripheral that offers programming-model accesses

The following figure shows a simplified SEMA42 block diagram. In the diagram, the register blocks named gate0, gate1,…, gate (n-2) and gate (n-1) include the finite state machines implementing the semaphore gates.

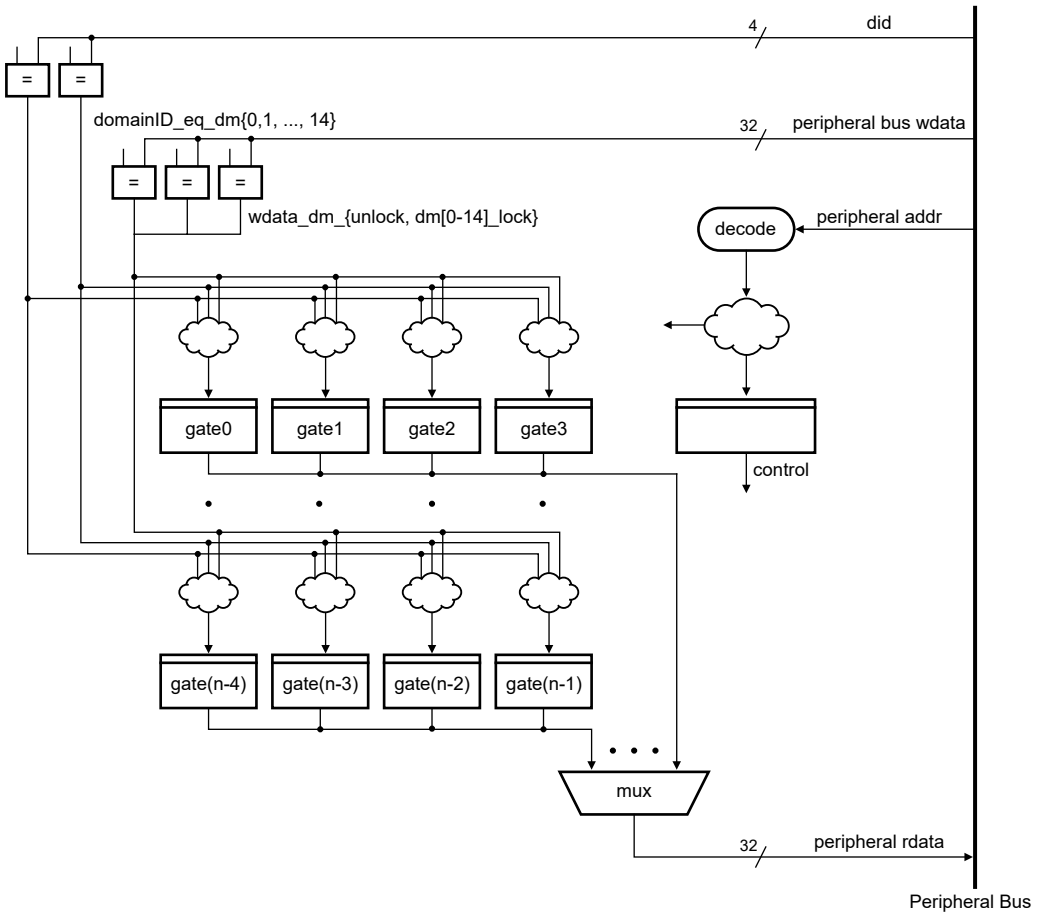


Figure 38. SEMA42 block diagram

16.3 Functional description

This section describes more about the SEMA42 functional operation and specific details of the state machines of the GATE*n* registers.

As described previously, each of the GATE*n* registers implements a 4-bit, 16-state machine. The following figure shows a simplified diagram of the state transitions for each gate.

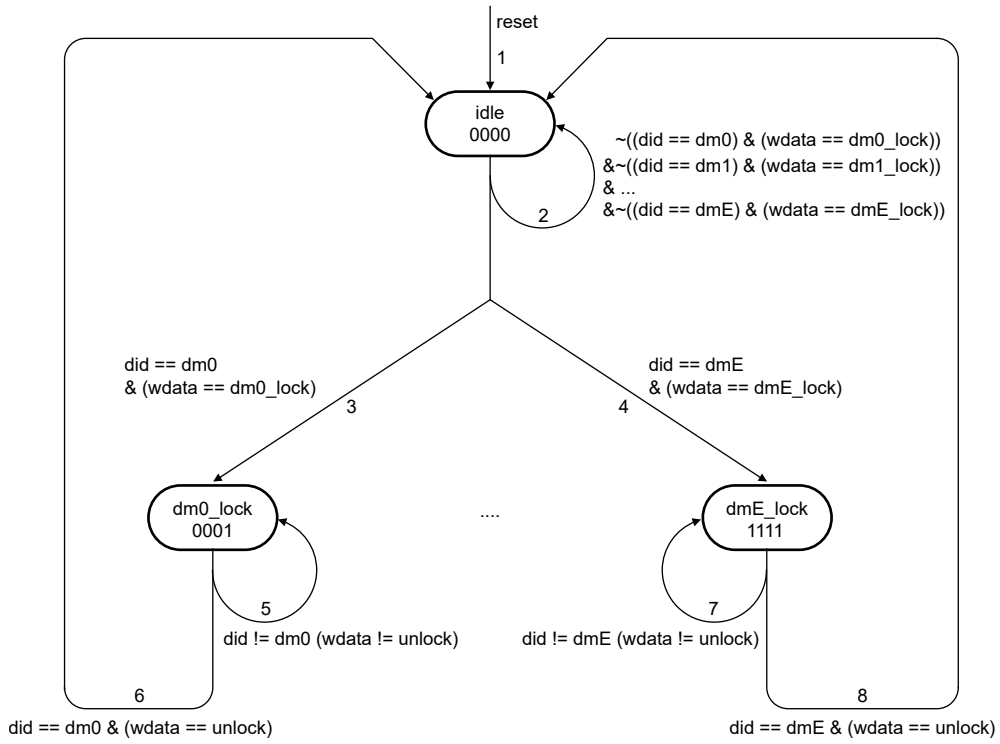


Figure 39. GATE*n* state machine

In the figure above, "did" is the domain identifier and "dm" is the domain number. The domain number identifies core domain *X* (dm*X*). Thus, for example, "dmE" represents domain 14 (E in hexadecimal). The platform passes the domain number to SEMA42.

The following table defines the GATE*n* state transitions.

Table 53. GATE*n* state transitions

Current state	Next state	Transition	Description
–	idle	1	Any reset, whether a system reset or a software-initiated gate reset, unconditionally forces the gate into the idle state.
idle	idle	2	Unless a write of the appropriate lock value from the corresponding domain occurs, the gate remains in the idle state.
idle	dm0_lock	3	When domain 0h initiates a write of the dm0_lock data value, the gate transitions into the dm0_lock state.
idle	dmE_lock	4	When domain Eh initiates a write of the dmE_lock value, the gate transitions into the dmE_lock state.
dm0_lock	dm0_lock	5	When in this state, the gate remains here if any attempted write is not from domain 0h with the unlock data value.
dm0_lock	idle	6	The gate returns to the idle (unlocked) state after a write from domain 0h with the unlock data value occurs.

Table continues on the next page...

Table 53. GATE*n* state transitions (continued)

Current state	Next state	Transition	Description
dmE_lock	dmE_lock	7	When in this state, the gate remains here if any attempted write is not from domain Eh with the unlock data value.
dmE_lock	idle	8	The gate returns to the idle (unlocked) state after a write from domain Eh with the unlock data value occurs.

SEMA42 uses these gate data values:

- The lock data value is (domain number) + 1.
- The unlock data value is 00h.

16.4 Memory map/register definition

You can access these registers only in Supervisor mode. User accesses terminate with an error.

16.4.1 SEMA42 register descriptions

16.4.1.1 SEMA42 memory map

SEMA42 base address: 4046_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h - Fh	Gate Register (GATE0 - GATE15) [1]	8	RW	00h
42h	Reset Gate Read (RSTGT_R)	16	R	0000h
42h	Reset Gate Write (RSTGT_W)	16	W	See section

1. In this array, the index and offset values of the registers do not increment in direct alignment. For details, see the register description.

16.4.1.2 Gate Register (GATE0 - GATE15)

Offset

For n = 0 to 15:

Register	Offset
GATEn	0h + (n + 3 - 2 × (n mod 4))

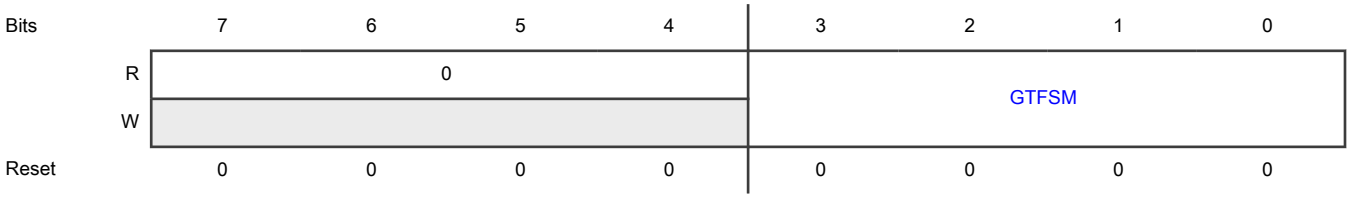
Function

SEMA42 implements each semaphore gate in a 4-bit finite state machine, right-justified in a byte data structure. The hardware uses the logical domain-identifier number in conjunction with the data patterns to validate all attempted write operations. Only domain masters can modify the gate registers. After a gate locks, only the locking domain must open (unlock) the gate.

You can read multiple gate values in a single access. However, you can update only a single gate at a time, via a write operation. If you attempt to write a byte-wide value that is neither the unlock value (00h) nor the appropriate lock value (domainID_number + 1),

SEMA42 considers this as "no operation" and does not change any gate state. Attempts to write multiple gates in a single-aligned access with a size larger than 8 bits (byte) generate an error termination and do not allow any gate state changes.

Diagram



Fields

Field	Function
7-4 —	Reserved
3-0 GTFSM	Gate finite state machine The state of the gate reflects the last domain that locked it. This can be useful during system debug. The hardware gate has a 16-state implementation, defined as: 0000b - The gate is unlocked (free). 0001b - Domain 0 locked the gate. 0010b - Domain 1 locked the gate. 0011b - Domain 2 locked the gate. 0100b - Domain 3 locked the gate. 0101b - Domain 4 locked the gate. 0110b - Domain 5 locked the gate. 0111b - Domain 6 locked the gate. 1000b - Domain 7 locked the gate. 1001b - Domain 8 locked the gate. 1010b - Domain 9 locked the gate. 1011b - Domain 10 locked the gate. 1100b - Domain 11 locked the gate. 1101b - Domain 12 locked the gate. 1110b - Domain 13 locked the gate. 1111b - Domain 14 locked the gate.

16.4.1.3 Reset Gate Read (RSTGT_R)

Offset

Register	Offset
RSTGT_R	42h

Function

This section and Reset Gate Write (RSTGT_W) describe the same register. This section describes the general operation of the register and also shows how the register fields appear when you read the register. Reset Gate Write (RSTGT_W) shows how the register fields appear when you write to the register.

The intent of the hardware gate implementation is to specify a protocol where the locking domain must unlock the gate. However, some systems may require a reset function to re-initialize the state of any gate(s) without requiring a system-level reset. To support this special gate reset requirement, SEMA42 implements a secure reset mechanism that allows you to initialize a hardware gate (or all the gates) by following a specific dual-write access pattern. The secure-gate reset:

- Uses a technique similar to that required for the servicing of a software watchdog timer
- Requires two consecutive writes with pre-defined data patterns from the same domain

You must do this to force the clearing of the specified gate(s). The required access pattern as follows:

1. A domain performs a 16-bit write to the RSTGT memory location. The most significant byte (RSTGT_W[RSTGDP]) must be E2h. The value of least significant byte is irrelevant for this reference and can be anything.
2. The same domain then performs a second 16-bit write to the RSTGT location. For this write, the upper byte (RSTGT_W[RSTGDP]) is the logical complement of the first data pattern (1Dh) and the lower byte (RSTGT_W[RSTGTN]) specifies the gate(s) to be reset. This gate field can specify a single gate or all gates to be cleared. If the same domain writes incorrect data on the second access or another domain performs the second write access, SEMA42 aborts the special gate reset sequence and does not assert an error signal.
3. Reads of the RSTGT location return information on the 2-bit reset state machine (RSTGT_R[RSTGSM]) that implements these functions:
   - The domain performing the reset (RSTGT_R[RSTGMS])
   - The last-cleared gate number(s) (RSTGT_R[RSTGTN])

   Reads of the RSTGT register do not affect the secure-reset finite state machine in any manner.

Diagram

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

Fields

Field	Function
15-14	ROZ

Table continues on the next page...

Table continued from the previous page...

Field	Function
ROZ	This field always returns the value 0 when you read it.
13-12 RSTGSM	Reset gate finite state machine Reads of the RSTGT register return the encoded state machine value. RSTGSM = 10b is valid for only a single machine cycle, so a read can never return this value. SEMA42 maintains the reset state machine in a 2-bit, 3-state implementation, defined as follows: 00b - Idle, waiting for the first data pattern write. 01b - Waiting for the second data pattern write 10b - The 2-write sequence has completed. Generate the specified gate reset(s). After the reset is performed, this machine returns to the idle (waiting for first data pattern write) state. 11b - This state encoding is never used and therefore reserved.
11-8 RSTGMS	Reset gate domain This field records the logical number of the domain performing the gate reset function. The logical number is the domain number. This domain number is determined by the XRDC's Master Domain Assignment Controller (XRDC_MDAC). To succeed, this function requires that the same domain initiate the two consecutive writes to this register. SEMA42 updates the field each time a write to this register occurs.
7-0 RSTGTN	Reset gate number This field specifies the specific hardware gate to be reset. The second write updates this field. If RSTGTN < 64, SEMA42 resets the single gate defined by RSTGTN. Otherwise, SEMA42 resets all the gates.

16.4.1.4 Reset Gate Write (RSTGT_W)

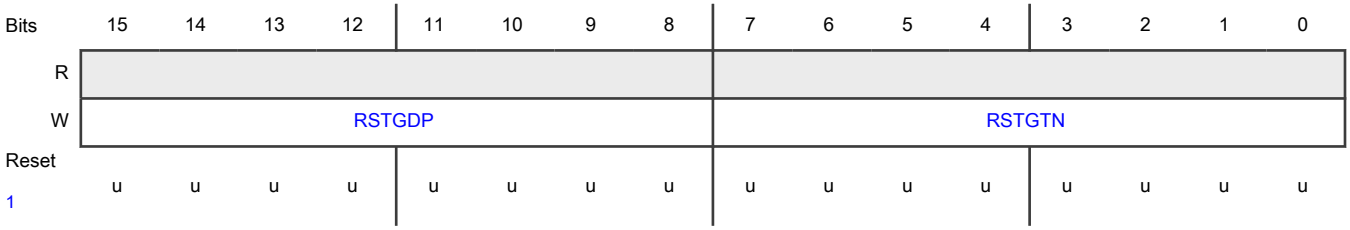
Offset

Register	Offset
RSTGT_W	42h

Function

This section describes how the RSTGT fields appear when the register is written. See Reset Gate Read (RSTGT_R) for the description of the register's overall operation and how the register fields appear when you read the register.

Diagram



1. Reset value is not applicable for writes.

Fields

Field	Function
15-8 RSTGDP	Reset gate data pattern You access this field with the specified data patterns on the two consecutive writes to enable the gate-reset mechanism. For the first write, RSTGDP must be E2h. For the second write, RSTGDP must be 1Dh.
7-0 RSTGTN	Reset gate number This field specifies the specific hardware gate to be reset. The second write updates this field. If RSTGTN < 64, SEMA42 resets the single gate defined by RSTGTN. Otherwise, SEMA42 resets all the gates.

16.5 Glossary

IPS Protocol used for peripheral accesses to the programming model

Chapter 17
Crossbar Integrity Checker (XBIC)

17.1 Chip-specific XBIC information

17.1.1 XBIC instances and configuration

This chip has up to seven instances of XBIC. The following tables describes the instances and their configuration.

Table 54. XBIC instances

Instance	MWCT2D17S/MWCT2D16S	MWCT2016S
XBIC_0	Yes	Yes
XBIC_1	Yes	No
XBIC_2	Yes	No
XBIC_3	Yes	No

17.1.2 Target master IDs

See "Chip-specific XRDC information" for master IDs.

17.2 Overview

XBIC verifies the integrity of crossbar transfers.

17.3 Features

XBIC includes the following features:

- Verification of attribute information for crossbar transfers [1],[2]
  - EDC detects single- and double-bit errors
- Verification of feedback information for each data phase during crossbar transfers
- Error injection for testing
  - Programmable master and slave port specifiers
  - Programmable 8-bit toggle vector to insert error in master EDC value
  - Address, EDC syndrome, master, and slave port information are captured when error is detected
- Crossbar transfer attribute integrity check programmable on a per-slave-port basis
- Feedback integrity check programmable on a per-master-port basis

17.4 Block diagram

The chip routes crossbar transfer attribute information for all mapped master and slave ports to XBIC, which calculates and checks the EDC value of the attribute information, as shown in the following diagram.

[1] See Table 55 for a list of crossbar attribute signals verified.
[2] The chip verifies read and write data separately, via the end-to-end ECC architecture.

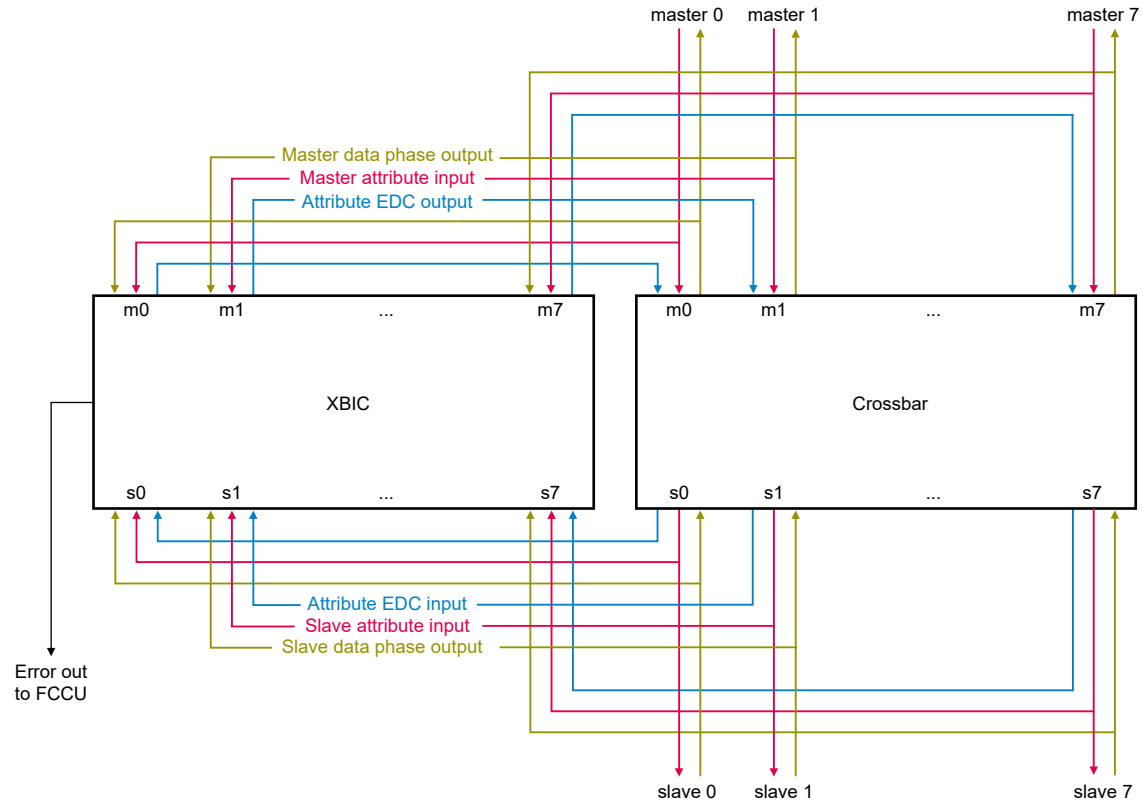


Figure 40. XBIC system block diagram

The above figure illustrates one of many possible XBIC and crossbar configurations. See the Chip-specific XBIC information section for actual port mappings.

17.5 Signal description

XBIC has no external interface signals.

17.6 Functional description

XBIC verifies the integrity of the crossbar interface on an individual port basis according to the configuration specified via the MCR register. When XBIC detects an error, it reports relevant information and sends an error signal to the FCCU module, but does not generate a bus error. XBIC integrity checking is independent of end-to-end ECC, which monitors the integrity of the transfer address and data.

During the address phase of a transfer, XBIC verifies the crossbar attribute information using an 8-bit EDC, which detects any single-bit or double-bit errors. When XBIC detects an error (an attribute integrity error), due to either hardware fault or error injection, it reports information related to the error in ESR and EAR.

During the data phase of a data transfer, XBIC verifies the integrity of response signals from slave to master as they pass through the crossbar. When XBIC detects an error in the response signals (a feedback integrity error), it reports the XBIC slave and master ports in ESR[DPSE0] - ESR[DPSE7] and ESR[DPME0] - ESR[DPME7], respectively.

During the data phase, XBIC sends an alarm to the FCCU module if a master port attempts back-to-back accesses in which:

1. The first access terminates normally.
2. The second attempts access to an address space not mapped to any slave on the crossbar.

The resultant 'absent slave error' causes the crossbar to generate a bus error response to the requesting master. XBIC detects the bus error response as a difference because the bus error did not originate from the slave port.

You can program XBIC to inject EDC errors for testing purposes. Error injection targets a single slave port and a single master port, as specified by the configuration settings in the EIR register. When XBIC inserts an error, it changes the EDC syndrome, causing the XBIC to assert an error indication to the FCCU module. Otherwise, transfers are unaffected by error injection. This enables verification of the check logic without compromising the integrity of the data transfer. After you enable error injection function by writing 1 to EIR[EIE], XBIC induces errors on all subsequent targeted transactions until you write a 0 to the field. After the FCCU error indication asserts, it remains asserted even after you write 0 to EIR[EIE]. The error indication deasserts after FCCU specifically clears the existing error. After FCCU clears the XBIC error, additional error injection testing can continue.

To trace a fault reported by the XBIC to the FCCU:

1. Note the error reported by the FCCU. For example, "NCF[46]".
2. Locate the source module and error description in the attached fault mapping spreadsheet. For example:
   - Channel number: NCF[46]
   - Source module: AXBS_1 integrity checker
   - Description: Instruction crossbar error indication to FCCU if syndrome calculated using EDC on the data is not zero
3. Refer to the Chip-specific XBIC information section for the source XBIC module to determine the specific XBIC module instance—"XBIC_1", for example.
4. Determine the type of error (attribute integrity error or feedback integrity error) and the XBIC port(s) involved by reading the information reported in the ESR register of the reported XBIC instance.
5. For attribute integrity check errors, read the XBIC EAR register for the target address of the requested transfer.
6. Refer to the Chip-specific XBIC information section and possibly the Chip-specific AXBS information section for port mapping of XBIC ports to AXBS ports.

Decode a single-bit error syndrome value reported in ESR[SYN] by finding the value in the following table. Any syndrome value not included in the table indicates a multi-bit error.

Table 55. Hexadecimal attribute single-bit error syndromes

Signal	SYN	Signal	SYN	Signal	SYN	Signal	SYN
hwrite	07	hbstrb[7]	70	hdecor[31]	25	hdecor[15]	23
htrans[0]	0b	hbstrb[6]	32	hdecor[30]	68	hdecor[14]	51
hsize[2]	0d	hbstrb[5]	52	hdecor[29]	c7	hdecor[13]	54
hsize[1]	0e	hbstrb[4]	a8	hdecor[28]	83	hdecor[12]	61
hsize[0]	13	hbstrb[3]	43	hdecor[27]	85	hdecor[11]	e3
hprot[5]	15	hbstrb[2]	45	hdecor[26]	86	hdecor[10]	e6
hprot[4]	16	hbstrb[1]	4c	hdecor[25]	89	hdecor[9]	f8
hprot[3]	19	hbstrb[0]	a4	hdecor[24]	8a	hdecor[8]	38
hprot[2]	1a	hmaster[3]	a2	hdecor[23]	8c	hdecor[7]	58
hprot[1]	1c	hmaster[2]	b0	hdecor[22]	49	hdecor[6]	37
hprot[0]	91	hmaster[1]	c1	hdecor[21]	92	hdecor[5]	f1

Table continues on the next page...

Table 55. Hexadecimal attribute single-bit error syndromes (continued)

Signal	SYN	Signal	SYN	Signal	SYN	Signal	SYN
hburst[2]	a1	hmaster[0]	c2	hdecor[20]	94	hdecor[4]	3b
hburst[1]	64	hslave[2]	c4	hdecor[19]	98	hdecor[3]	3d
hburst[0]	29	hslave[1]	c8	hdecor[18]	46	hdecor[2]	3e
hmastlock	2a	hslave[0]	d0	hdecor[17]	34	hdecor[1]	4f
hunalign	2c	hdecorated	e0	hdecor[16]	4a	hdecor[0]	6e
edc[7]	80	edc[6]	40	edc[5]	20	edc[4]	10
edc[3]	08	edc[2]	04	edc[1]	02	edc[0]	01

17.7 XBIC register descriptions

The XBIC programming model consists of four 32-bit registers. Software can access this model only in supervisor mode using 32-bit (word) accesses. Each of the following generates a transaction error back to the requesting master. Such errors could cause core exceptions apart from other problems.

- Access size other than 32-bit
- Access to an undefined (reserved) address
- Unsupported access type—an attempted write to a read-only register, for example
- Access in user mode

17.7.1 XBIC memory map

XBIC_AXBS base address: 4020_4000h

XBIC_AXBS_PERI base address: 4020_8000h

XBIC_AXBS_TCM base address: 4040_0000h

XBIC_AXBS_eDMA base address: 4040_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	XBIC Module Control (MCR)	32	RW	FFFF_0000h
4h	XBIC Error Injection (EIR)	32	RW	0000_0000h
8h	XBIC Error Status (ESR)	32	R	0000_0000h
Ch	XBIC Error Address (EAR)	32	R	0000_0000h

17.7.2 XBIC Module Control (MCR)

Offset

Register	Offset
MCR	0h

Function

Use this register to turn attribute integrity checking and feedback integrity checking on or off on a per-port basis.

- Turn on attribute integrity checking on one or more XBIC slave ports by ensuring that the associated SE*n* field(s) have a value of 1. For example, setting field SE*0* enables attribute integrity checking for slave port 0. The default (reset) behavior is for attribute integrity checking to be performed for all slave ports. XBIC performs EDC-based checks on all transfer requests targeting the selected slave port(s). The signals verified are transfer attribute signals going from master to slave. When XBIC detects an attribute integrity error, it reports relevant information in the ESR and EAR registers.
- Turn on feedback integrity checking on one or more XBIC master ports by ensuring that the associated ME*n* field(s) have a value of 1. For example, setting field ME*0* to 1 enables feedback integrity checking for master port 0. The default (reset) behavior is for feedback integrity checking to be performed for all master ports. XBIC checks slave-to-master feedback signals for transfer requests originating from the selected XBIC master port(s). If any feedback signal value is different at the master and slave ports during the data phase, XBIC reports the relevant master and slave ports in the ESR register.

Each field in this register references a specific master or slave port using XBIC port numbering. Referring to Figure 40, "slave port 0" refers to XBIC port "s0" in the figure, "slave port 1" refers to port "s1", and so on. Similarly, "master port 0" refers to XBIC port "m0", "master port 1" refers to port "m1", and so on. See the "Chip-specific XBIC information" section in this document for the mapping of XBIC instances to AXBS instances and XBIC ports to AXBS ports. See the "Chip-specific AXBS information" section in this document for the device component(s) mapped to each port of an AXBS instance.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SE0	SE1	SE2	SE3	SE4	SE5	SE6	SE7	ME0	ME1	ME2	ME3	ME4	ME5	ME6	ME7
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Fields

Field	Function
31 SE0	Slave port EDC Error Detection Enable 0b - Attribute integrity checking disabled for slave port 0 1b - Attribute integrity checking enabled for slave port 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 SE1	Slave port EDC Error Detection Enable 0b - Attribute integrity checking disabled for slave port 1 1b - Attribute integrity checking enabled for slave port 1
29 SE2	Slave port EDC Error Detection Enable 0b - Attribute integrity checking disabled for slave port 2 1b - Attribute integrity checking enabled for slave port 2
28 SE3	Slave port EDC Error Detection Enable 0b - Attribute integrity checking disabled for slave port 3 1b - Attribute integrity checking enabled for slave port 3
27 SE4	Slave port EDC Error Detection Enable 0b - Attribute integrity checking disabled for slave port 4 1b - Attribute integrity checking enabled for slave port 4
26 SE5	Slave port EDC Error Detection Enable 0b - Attribute integrity checking disabled for slave port 5 1b - Attribute integrity checking enabled for slave port 5
25 SE6	Slave port EDC Error Detection Enable 0b - Attribute integrity checking disabled for slave port 6 1b - Attribute integrity checking enabled for slave port 6
24 SE7	Slave Port EDC Error Detection Enable 0b - Attribute integrity checking disabled for slave port 7 1b - Attribute integrity checking enabled for slave port 7
23 ME0	Master Port Enable For Feedback Integrity Check 0b - Feedback integrity checking disabled for master port 0 1b - Feedback integrity checking enabled for master port 0
22 ME1	Master Port Enable For Feedback Integrity Check 0b - Feedback integrity checking disabled for master port 1 1b - Feedback integrity checking enabled for master port 1
21 ME2	Master Port Enable For Feedback Integrity Check 0b - Feedback integrity checking disabled for master port 2 1b - Feedback integrity checking enabled for master port 2
20 ME3	Master Port Enable For Feedback Integrity Check

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Feedback integrity checking disabled for master port 3 1b - Feedback integrity checking enabled for master port 3
19 ME4	Master Port Enable For Feedback Integrity Check 0b - Feedback integrity checking disabled for master port 4 1b - Feedback integrity checking enabled for master port 4
18 ME5	Master Port Enable For Feedback Integrity Check 0b - Feedback integrity checking disabled for master port 5 1b - Feedback integrity checking enabled for master port 5
17 ME6	Master Port Enable For Feedback Integrity Check 0b - Feedback integrity checking disabled for master port 6 1b - Feedback integrity checking enabled for master port 6
16 ME7	Master Port Enable For Feedback Integrity Check 0b - Feedback integrity checking disabled for master port 7 1b - Feedback integrity checking enabled for master port 7
15-0 —	Reserved

17.7.3 XBIC Error Injection (EIR)

Offset

Register	Offset
EIR	4h

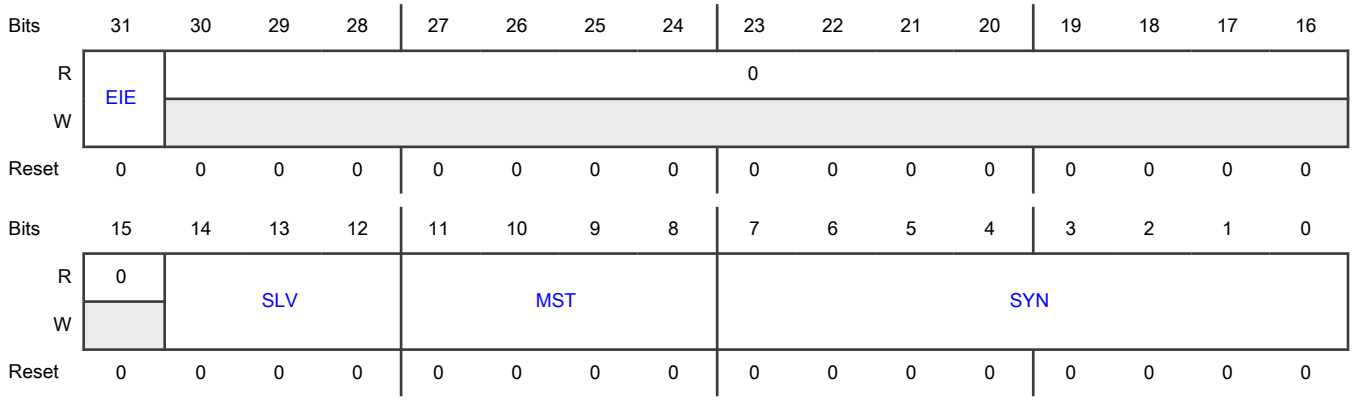
Function

Use this register to configure the XBIC error injection function and turn it on or off. When enabled, the XBIC error injection function inserts an attribute integrity error when the targeted master requests a transaction of the targeted slave port. The inserted error changes the calculated EDC syndrome value, causing XBIC to:

- Capture transfer information in the ESR and EAR registers
- Assert an error signal to the FCCU module

Otherwise, XBIC error injection does not affect transfers.

Diagram



Fields

Field	Function
31 EIE	Error Injection Enable 0b - Error injection disabled 1b - Error injection enabled
30-15 —	Reserved
14-12 SLV	Target Slave Port Specifies the target slave port for error injection—other slave ports are unaffected. Specify the target slave port by its XBIC slave port number (0–7). See the "Chip-specific XBIC information" section in this document for the mapping of XBIC instances to AXBS instances and XBIC ports to AXBS ports. See the "Chip-specific AXBS information" section in this document for the device component(s) mapped to each port of an AXBS instance.
11-8 MST	Target Master ID Specifies the target master port for error injection—transfers with other masters are unaffected. Specify the target master port using the logical master ID number of the target bus master. See the "Chip-specific XBIC information" section in this document for the master IDs and their corresponding components.
7-0 SYN	Syndrome XBIC performs an exclusive OR operation on the value in this field and the calculated syndrome, to generate an error with the specified syndrome. A value of zero does not generate an error. See Table 55 for a list of transfer attribute single-bit error syndromes, noting that the values given in the table are hexadecimal.

17.7.4 XBIC Error Status (ESR)

Offset

Register	Offset
ESR	8h

Function

In this register, XBIC reports information about the most recent transfer with an error detected. If XBIC detects an attribute integrity check error, it reports:

- The slave port identifier (SLV)
- The master port identifier (MST)
- The error syndrome (SYN)

If XBIC detects a mismatch among feedback signals during the data phase:

- The DPSE0 - DPSE7 field with a value of 1 indicates the XBIC slave port. In the DPSE0-DPSE7 field descriptions, the slave port number refers to the XBIC slave port number. Referring to Figure 40, "slave port 0" refers to XBIC port "s0" in the figure, "slave port 1" refers to port "s1", and so on. See the "Chip-specific XBIC information" section in this document for the mapping of XBIC instances to AXBS instances and XBIC ports to AXBS ports. See the "Chip-specific AXBS information" section in this document for the device component(s) mapped to each port of an AXBS instance.
- The DPME0 - DPME7 field with a value of 1 indicates the XBIC master port. In the DPME0-DPME7 field descriptions, the master port number refers to the XBIC master port number. Referring to Figure 40, "master port 0" refers to XBIC port "m0" in the figure, "master port 1" refers to port "m1", and so on. See the "Chip-specific XBIC information" section in this document for the mapping of XBIC instances to AXBS instances and XBIC ports to AXBS ports. See the "Chip-specific AXBS information" section in this document for the device component(s) mapped to each port of an AXBS instance.

XBIC sets this register to all 0s only on reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	DPSE 0	DPSE 1	DPSE 2	DPSE 3	DPSE 4	DPSE 5	DPSE 6	DPSE 7	DPME 0	DPME 1	DPME 2	DPME 3	DPME 4	DPME 5	DPME 6
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31 VLD	Error Status Valid 0b - No error detected—other fields of the ESR and EAR registers are invalid 1b - Error detected—all fields of the ESR and EAR registers are valid
30 DPSE0	Data Phase Slave Port Error 0b - No feedback integrity error detected on slave port 0 1b - Feedback integrity error detected on slave port 0
29 DPSE1	Data Phase Slave Port Error 0b - No feedback integrity error detected on slave port 1 1b - Feedback integrity error detected on slave port 1
28 DPSE2	Data Phase Slave Port Error 0b - No feedback integrity error detected on slave port 2 1b - Feedback integrity error detected on slave port 2
27 DPSE3	Data Phase Slave Port Error 0b - No feedback integrity error detected on slave port 3 1b - Feedback integrity error detected on slave port 3
26 DPSE4	Data Phase Slave Port Error 0b - No feedback integrity error detected on slave port 4 1b - Feedback integrity error detected on slave port 4
25 DPSE5	Data Phase Slave Port Error 0b - No feedback integrity error detected on slave port 5 1b - Feedback integrity error detected on slave port 5
24 DPSE6	Data Phase Slave Port Error 0b - No feedback integrity error detected on slave port 6 1b - Feedback integrity error detected on slave port 6
23 DPSE7	Data Phase Slave Port Error 0b - No feedback integrity error detected on slave port 7 1b - Feedback integrity error detected on slave port 7
22 DPME0	Data Phase Master Port Error 0b - No feedback integrity error detected on master port 0 1b - Feedback integrity error detected on master port 0
21	Data Phase Master Port Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
DPME1	0b - No feedback integrity error detected on master port 1 1b - Feedback integrity error detected on master port 1
20 DPME2	Data Phase Master Port Error 0b - No feedback integrity error detected on master port 2 1b - Feedback integrity error detected on master port 2
19 DPME3	Data Phase Master Port Error 0b - No feedback integrity error detected on master port 3 1b - Feedback integrity error detected on master port 3
18 DPME4	Data Phase Master Port Error 0b - No feedback integrity error detected on master port 4 1b - Feedback integrity error detected on master port 4
17 DPME5	Data Phase Master Port Error 0b - No feedback integrity error detected on master port 5 1b - Feedback integrity error detected on master port 5
16 DPME6	Data Phase Master Port Error 0b - No feedback integrity error detected on master port 6 1b - Feedback integrity error detected on master port 6
15 DPME7	Data Phase Master Port Error 0b - No feedback integrity error detected on master port 7 1b - Feedback integrity error detected on master port 7
14-12 SLV	Slave Port Slave port targeted by the most recent transfer with an attribute integrity check error detected. The value in this field is the XBIC slave port number (0–7). See the "Chip-specific XBIC information" section in this document for the mapping of XBIC instances to AXBS instances and XBIC ports to AXBS ports. See the "Chip-specific AXBS information" section in this document for the device component(s) mapped to each port of an AXBS instance.
11-8 MST	Master ID Master port that requested the most recent transfer with an attribute integrity check error detected. The value in this field is the logical master ID number of the bus master. See the "Chip-specific XBIC information" section in this document for the master IDs and their corresponding components.
7-0 SYN	Syndrome Syndrome calculated for the most recent transfer with an attribute integrity check error detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For single-bit errors, identify the signal in error by matching the SYN value in Table 55, noting that the syndrome (SYN) values in the table are hexadecimal.

17.7.5 XBIC Error Address (EAR)

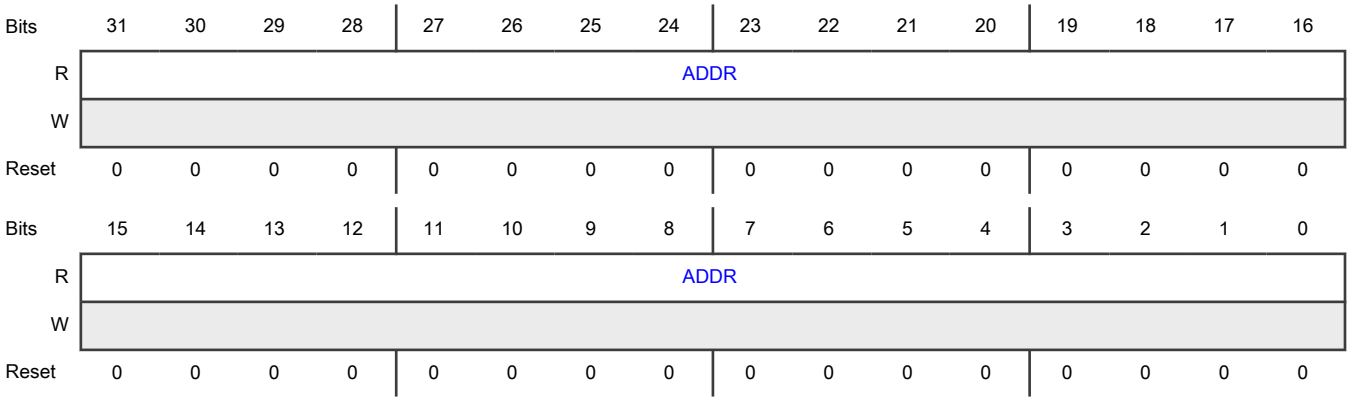
Offset

Register	Offset
EAR	Ch

Function

In this register, XBIC reports the address of the most recent transfer with an attribute integrity check error detected—either because of a hardware fault or error injection. XBIC sets this register to all 0s only on reset.

Diagram



Fields

Field	Function
31-0 ADDR	Error Address Address of the most recent transfer with an attribute integrity check error detected.

17.8 Glossary

EDC	Error Detection Code
hdecor[31:0]	Non-standard AHB address phase signal for transporting optional decorated storage instruction information
hdecorated	Non-standard AHB address phase signal for transporting optional decorated storage instruction information

Chapter 18
Extended Resource Domain Controller (XRDC)

18.1 Chip-specific XRDC information

18.1.1 MDAC configuration

All MDACs with DID = 0 use the default DID parameter.

Table 56. MDAC configuration

Submodule instance	Configuration	XRDC MDACFG#	Bus master	Master ID value	Default DID	PID	Nonsecure input	Applicability
XRDC_MDAC0	Processor	1	Cortex-M7_0, AXI, AHBP Cortex-M7_0 debug	0h	0h	PID0	0b	MWCT2015S/ MWCT2016S/ MWCT2D17S/ MWCT2D16S
XRDC_MDAC1	Nonprocessor	1	eDMA AHB	2h	0h	—	1b	All
XRDC_MDAC4	Processor	1	Cortex-M7_1, AXI, AHBP Cortex-M7_1 debug	1h	0h	PID4	0b	MWCT2D17S, MWCT2D16S
XRDC_MDAC5	Nonprocessor	1	EMAC AHB	4h	0h	—	1b	MWCT2D17S, MWCT2D16S

18.1.2 MRC configuration

Table 57. MRC configuration

Submodule instance	Region format	Number of region descriptors	Slaves protected (port number)	Applicability
XRDC_MRC0	Auto	16[1] or 8	PFLASH_0 (0) PFLASH_1 (1) PFLASH_2 (2)[1] PFLASH_3 (3) PFLASH_WR (3)	MWCT2015S, MWCT2016S, MWCT2D16S, MWCT2D17S
XRDC_MRC1	Auto	16[1] or 8	PRAM0_0 (0) PRAM1_0[1] (2) TCM backdoor (1)	MWCT2015S, MWCT2016S, MWCT2D16S, MWCT2D17S
XRDC_MRC2	Auto	4	QuadSPI (0)	MWCT2D16S, MWCT2D17S

1. Applicable to MWCT2D16S and MWCT2D17S

18.1.3 PAC configuration

Table 58. PAC configuration

Module name	Slaves protected	Applicability
XRDC_PAC0	AIPS_0	MWCT2015S, MWCT2016S, MWCT2D16S, MWCT2D17S
XRDC_PAC1	AIPS_1	MWCT2015S, MWCT2016S, MWCT2D16S, MWCT2D17S
XRDC_PAC2	AIPS_2	MWCT2D16S, MWCT2D17S

NOTE

For PDAC registers assignment to peripherals, see the memory map file attached to this document.

18.1.4 Domain Error Word registers (DERR_W*x*_0-18) mapping

The mapping of the domain error capture registers is as follows:

Table 59. Domain Error Word registers mapping

Register	Corresponding MRC/PAC	Available in chips
DERR_W*x*_0	MRC0	MWCT2015S, MWCT2016S, MWCT2D16S, MWCT2D17S
DERR_W*x*_1	MRC1	MWCT2015S, MWCT2016S, MWCT2D16S, MWCT2D17S
DERR_W*x*_2	MRC2	MWCT2D16S, MWCT2D17S
DERR_W*x*_16	PAC0	MWCT2015S, MWCT2016S, MWCT2D16S, MWCT2D17S
DERR_W*x*_17	PAC1	MWCT2015S, MWCT2016S, MWCT2D16S, MWCT2D17S
DERR_W*x*_18	PAC2	MWCT2D16S, MWCT2D17S

Where *x*: 0, 1, 2, 3.

If the above registers are accessed in chips wherein they are not present, a bus error gets reported.

18.1.5 Exceptions and violations

A write attempt by a noncore bus master outside the defined ranges leads to an exception in case the XRDC region is defined to prevent noncore master access. The chip generates a bus error if you violate XRDC policies.

18.1.6 Configuration using SBAF

SBAF must protect access to its own resources. Hence, XRDC is configured during initialization. SBAF provides the mechanism for you to configure XRDC during boot. You must program the configuration data in the application flash memory region.

18.1.6.1 Default configuration of MDAC when HSE_B firmware feature is disabled

Table 60. MDAC registers default value

MDAC register	Value	Domain number assigned
HSE_B CPU (MDAC3)	C000_0002h	0

All the other MDAC domain numbers are 0 and SBAF does not lock them.

18.1.6.2 Default configuration of MRC registers when HSE_B firmware usage feature flag is disabled

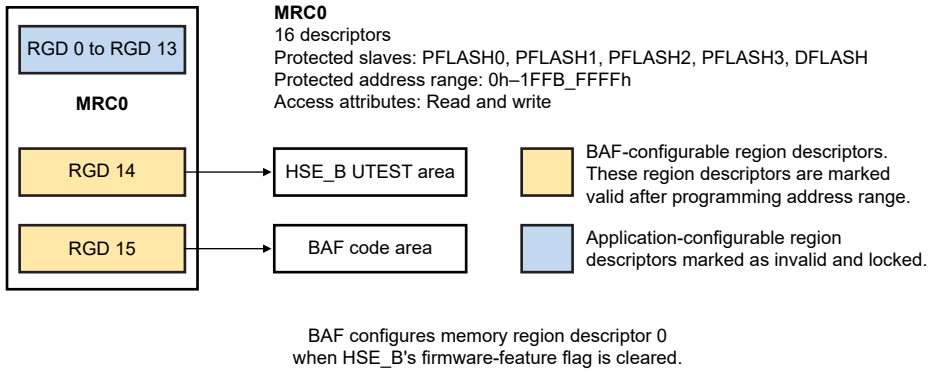


Figure 41. Default configuration of MRC registers when HSE_B firmware usage feature flag is disabled

Table 61. Default configuration of MRC registers when HSE_B firmware usage feature flag is disabled

MRC number	Region descriptor number	Region address range (in hex)	Name	Remarks	SBAF's configuration
0	14	1B00_0200 - 1B00_06FF	HSE_B UTEST area	UTEST area that is accessible by HSE_B core only. Start address is 1B00_0200h and 1B00_06FFh for others LC	W0 = start address = 1B00_0200h W1 = end address = 1B00_06FFh W2 = {SE = 0, SNUM = 0, D3ACP = 7h [all permission]}
1	15	007F_4000 - 007F_FFFF	SBAF code area	Code area for SBAF and HSE_B firmware	W0 = start address= 007F_4000h W1 = end address= 007F_FFFFh W2 = {SE = 0, SNUM = 0, D2ACP =7h [all permission], D1ACP = 0h [no permission], D0ACP = 0h [no permission]}

18.1.6.3 PDAC default configuration

SBAF configures and lock the following peripherals for its exclusive use. Read and write access to these peripheral are not provided to application domains.

Table 62. PDAC default configuration

Peripheral	Peripheral PDAC number	Remarks
Flash memory controller alternate	155	HSE_B uses the alternate interface exclusively.
Flash memory alternate	188	HSE_B uses the alternate interface exclusively.
HSE_GPR	231	All cores have all permissions.

18.1.7 Configuration when HSE_B firmware-feature flag is cleared

When the HSE_B firmware-feature flag is cleared, SBAF:

- Does not permit XRDC configuration.
- Locks the aforementioned configurations.

18.2 Overview

XRDC manages access control between masters (cores and noncore masters) and targets (memories and peripherals) by placing them in virtual groups called domains.

Conceptually, a domain is one or more masters and memories and peripherals, that are isolated from others. It may help to look at a domain as a permissions group within a computing environment. All masters in a domain have the same access to chip resources such as memory and peripherals. See Introduction to domains for more information on domains.

The protection provided by XRDC access control is in addition to the local memory protection unit contained within each core.

18.2.1 Features

- Enables you to partition chip resources (master and target) into access-controlled domain .
  - Each domain has a unique DID .
  - The DID is an attribute of every system bus transaction.
- Provides a four-level hierarchical access control scheme for defining an ACP for each target in a domain. See Access control model for more information.
  - Memory region descriptors define access policies for address ranges within memories.
  - Peripheral access control registers define access policies for individual peripherals.
- Supports optional hardware semaphores to dynamically modify access rights for target resources.

18.2.2 Block diagram

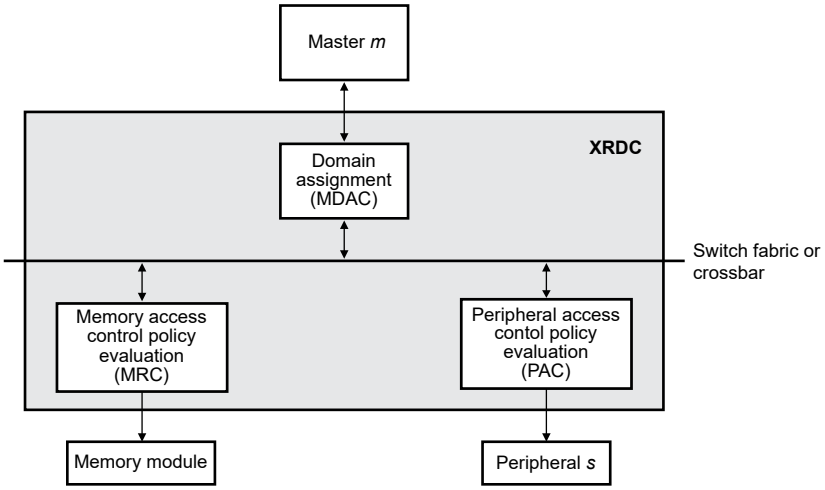


Figure 42. Block diagram

18.2.3 Block descriptions

Block	Description
Domain assignment	A process that adds information to transactions, including: • DID • Privileged attribute

Table continues on the next page...

Table continued from the previous page...

Block	Description
	• Secure attribute Domain assignment is performed by the MDAC submodule. See: • Domain assignment • Master domain assignment controller (MDAC)
Master	A core or noncore (for example, DMA) module that can initiate transactions with memory or peripheral resources.
Memory	A block of flash memory, RAM, or other memory.
Memory access control policy evaluation	A process that determines whether the domain associated with a transaction has access rights to a memory location. The process is performed by the MRC submodule. See: • Memory region ACP evaluation • Memory region controller (MRC)
Peripheral	A nonmemory resource module within the chip—an ADC, timer module, or communications module, for example.
Peripheral access control policy evaluation	A process that determines whether the domain associated with a transaction has access rights to a peripheral. The process is performed by the PAC submodule. See Peripheral access controller (PAC).
Crossbar	The chip's module and I/O interconnect infrastructure.

18.2.4 Indexes used in this chapter

Table 63. Indexes used in this chapter

Index	Description
c	Memory controller number. For example, MRC*c*.
d	Domain number. For example, PDAC_W0_6[D*d*ACP].
m	Master number. For example, PID*m*.
r	Memory region number. For example, RGD_W0_*r*.
s	PDAC slot number. For example, PDAC_W0_*s*[D0ACP].
w	Word number. In a group of registers consisting of consecutive 32-bit registers, *w* is the 0-indexed register number. For example, MRGD_W*w*_0.

18.2.5 Exceptions and violations

A write attempt by a noncore bus master outside the defined ranges leads to an exception in case the XRDC region is defined to prevent noncore master access. The chip generates a bus error if you violate XRDC policies.

18.3 Modes of operation

XRDC does not support any special modes of operation.

18.4 External signal description

XRDC does not have any external signals.

18.5 Functional description

18.5.1 Introduction to domains

A domain typically consists of one or more master, along with the memories and peripherals those masters are allowed to access. Because it is access-controlled, a domain acts as an independent computing environment.

Domains enable applications to coexist on the same silicon with a firewall between them, enforcing absolute interference protection.

Generally, you create each domain to meet a specific need. Examples of XRDC domain usage include:

- Isolation of real-time from non-real-time applications to ensure resource availability
- Isolation of safety-critical code from non-safety-critical code
- Isolation of third-party untrusted applications from trusted software
- Isolation of memory regions to ensure data security or to prevent accidental overwrites
- Limiting write access for a specific area of system memory to a single DMA module instance
- Limiting read access for a specific area of system memory to a specific processor core

You can assign a core to multiple domains but only one of those domains can be active at a given time.

Within each XRDC instance, each domain has a unique identifier, known as its DID. If an XRDC instance has 16 DIDs, that means the instance has 16 available domains.

You control which masters can access a peripheral by configuring the domain ACP for each peripheral. Similarly, you control which masters can access a memory region by configuring the ACP for each memory region.

18.5.2 Submodules

XRDC implements its functionality through its hardware submodules:

- Master domain assignment controller (MDAC)
- Memory region controller (MRC)
- Peripheral access controller (PAC)

18.5.2.1 Master domain assignment controller (MDAC)

The MDAC submodule performs domain assignment logic. XRDC contains an MDAC submodule for each XRDC-protected master. Each MDAC assigns a DID to every transaction from its associated master. You configure the domain assignment activity for each MDAC through a set of registers, in one of two formats:

- DFMT0 core domain assignment registers
- DFMT1 noncore domain assignment registers

To understand the role of domains in XRDC protection, see Introduction to domains.

For a full explanation of the domain assignment process, see Domain assignment.

18.5.2.2 Memory region controller (MRC)

The MRC submodule performs memory region access control. Each XRDC instance contains the number of MRCs indicated in HWCFG0[NMRC]. Each MRC is associated with one or more memories (see the chip-specific XRDC information for details). The MRC controls memory access using memory region descriptors. MRCFG*c*[NMRGD] indicates the number of region descriptors available for MRC *c*.

Each memory region descriptor defines a memory address range and a configurable access control policy for each domain using a set of four or five 32-bit registers (see Memory ACP evaluation registers).

Memory region descriptors also support including an optional hardware semaphore in the ACP evaluation for memory regions shared by multiple domains (see Hardware semaphores and dynamic access rights).

For a full explanation of the memory region ACP evaluation process, see Memory region ACP evaluation.

18.5.2.3 Peripheral access controller (PAC)

The PAC submodule provides domain access control for all peripherals connected to a single peripheral bus. Each XRDC contains the number of PACs indicated in HWCFG0[NPAC]. Each PAC supports up to 128 peripheral slots (see Finding the PDAC slot number for a peripheral). You configure the ACP for each peripheral using a set of PDAC_W*w_s* registers (see Peripheral ACP evaluation registers).

Peripheral access control also enable a hardware semaphore to be included in the access control policy evaluation for peripherals shared by multiple domains. See Hardware semaphores and dynamic access rights for more details.

For a full explanation of the peripheral ACP evaluation process, see Peripheral ACP evaluation process.

18.5.3 Transaction protection

During application execution, high-level chip modules such as cores or DMA, Ethernet, Zipwire, or FlexRay, known as masters, initiate transaction requests to memory and peripheral resources. XRDC adds protection capabilities to ensure the requesting master accesses only the resources that it is authorized to access. These capabilities support security and safety requirements.

XRDC provides this protection by adding two steps to the unprotected transaction flow, as shown in XRDC transaction flow.

XRDC transaction processing differs depending on:

- The type of master making the request (see Transaction request sources).
- The type of target receiving the request (see Transaction targets).

NOTE

See chip-specific XRDC information for the domain ID of each master on chip.

18.5.4 XRDC transaction flow

Table 64. XRDC transaction flow

Step	Operation	Performed by	Description
1	Transaction request	Master	A master requests a read or write transaction targeting memory or a peripheral.
2	Domain assignment	XRDC	XRDC MDAC submodule intercepts the request and performs domain assignment, which adds this metadata to the transaction: • DID • Privileged attribute • Secure attribute

Table continues on the next page...

Table 64. XRDC transaction flow (continued)

Step	Operation	Performed by	Description
3	Interconnect	Chip	The chip transmits the domain-assigned transaction request across the interconnect (crossbar).
4	ACP evaluation	XRDC	XRDC MRC or PAC submodule intercepts the transaction request and evaluates it against the target's ACP to determine whether the requesting master has sufficient access rights to the target. If it does, the transaction continues. Otherwise, XRDC generates an access violation error and the transaction terminates.
5	Transaction response	Target resource	If the previous step does not generate an access violation error, the target resource processes the transaction request and transmits data (for read transactions) and transaction status information (for all transactions) back across the interconnect to the requesting master. XRDC is not involved in the flow of information from the target resource back to the requesting master.

18.5.5 Domain assignment

18.5.5.1 Overview

Domain assignment associates a DID with each transaction request from a master. To determine the DID for a transaction, XRDC evaluates the domain-specific configuration data in the set of MDAC registers (MDA_W*n*_*m*_DFMT0 or MDA_W*n*_*m*_DFMT1) associated with the requesting master. The exact process depends on the source of the request (see Transaction request sources).

If the value of MDACFG*m*[NMDAR] is 1, which means a single W*w* register for a given master, the specified domain identifier is used directly. In case this value is more than 1, which means there are multiple W*w* registers for a given master, MDAC evaluates the conditional terms to determine a "hit". For all W*w* hits, their corresponding domain identifiers are logically summed together (boolean OR). Use cases are typically expected to hit in a single W*w*. Special care is needed if none of the conditional terms hit in any Wn evaluation; in this case, the generated DID = 0 and you must be aware of any potential access rights granted for this DID.

The number of MDAC registers can vary for each master. See the chip-specific information. MDACFG*m*[NMDAR] indicates the number of MDA registers, where *m* is the master number. You need *m* to locate the registers relevant for domain assignment. See the chip-specific XRDC information for more on master numbers. See:

- Register settings for DFMT0 direct domain assignment transactions
- Register settings for DFMT0 PID-based transactions
- Register settings for DFMT1 direct domain assignment transactions

Domain assignment also assigns the secure and privileged attributes to the transaction.

18.5.5.2 PID-based domain assignment

To provide more flexibility in routing core tasks to chip resources in different access-controlled domains, XRDC supports the use of a PID. If the core master contains a built-in PID register, indicated by HWCFG2[PIDP*m*] = 1 , XRDC reflects the core PID value in PID*m*[PID], and bit 5 of that field indicates the secure attribute for the transaction request. Otherwise, an application can mimic PID-based domain assignment by writing a value to that field.

18.5.5.3 Transaction request sources

The domain assignment process for an XRDC-protected transaction request depends on the source of the request.

Table 65. Transaction request sources

Request source	Topic	Brief description
Core master	DFMT0 direct domain assignment example	Direct domain assignment using a DFMT0 master domain assignment register.
Core master	DFMT0 PID-based domain assignment example	PID-based domain assignment using a DFMT0 master domain assignment register.
Core master	DFMT1 direct domain assignment example-single MDA	Direct domain assignment using a DFMT1 master domain assignment register with single MDA.

18.5.5.4 DFMT0 core domain assignment registers

Table 66. DFMT0 core domain assignment registers

Register	Index
MDACFG*m*	*m* = master number. See the chip-specific XRDC information for the available list of masters with their IDs.
MDA_W*w*_*m*_DFMT0[1]	*m* = master number. *w* = word (see MDA register structure). MDACFG*m*[NMDAR] indicates the number of MDA registers (words) per master.
PID*m*	*m* = master number.

1. See MDA register structure.

18.5.5.5 DFMT1 noncore domain assignment registers

Table 67. DFMT1 noncore domain assignment registers

Register	Index
MDACFG*m*	*m* = master number. See the chip-specific XRDC information for the available list of masters and their IDs.
MDA_W*w*_*m*_DFMT1[1]	*m* = master number. *w* = word. MDACFG*m*[NMDAR] indicates the number of MDA registers (words) per master.

1. See MDA register structure.

18.5.5.6 MDA register structure

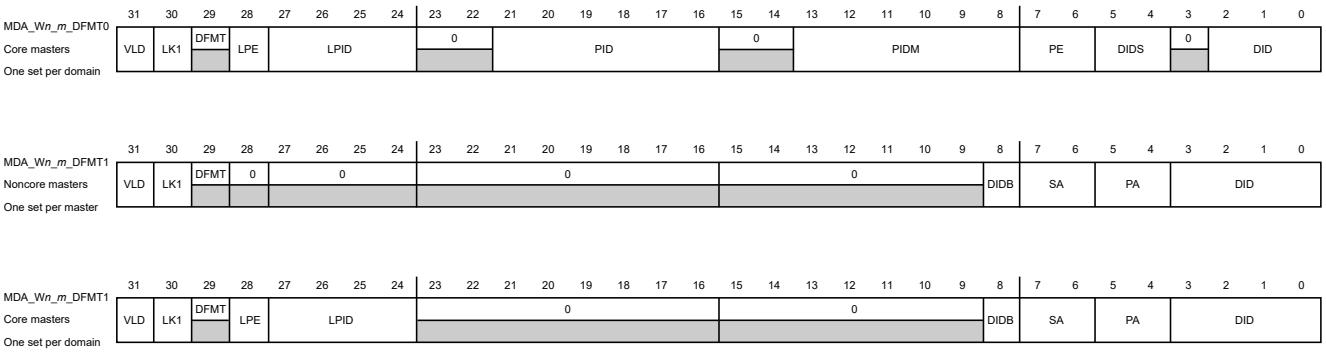


Figure 43. MDA register structure

NOTE

This XRDC configuration does not have LPE and LPID fields and should be considered as reserved fields.

18.5.6 ACP evaluation

18.5.6.1 Overview

When XRDC is not enabled, all peripherals and memories allow unrestricted access. XRDC allows you to limit that access to requests from a particular domain or domains with an application-specified ACP. XRDC intercepts the transaction request and evaluates it against the target's ACP to determine whether the requesting master has sufficient access rights to the target, based on the:

- Domain ID assignment (DID)
- Privileged attribute
- Secured attribute

XRDC obtains the target resource's domain ACP from the associated Domain Access Control Policy (D*d*ACP) field (see Domain ACP specification) in the appropriate register set:

- Peripheral ACP evaluation registers
- Memory ACP evaluation registers

If ACP evaluation determines that the transaction request has sufficient access rights to the target resource, XRDC allows the transaction to continue. Otherwise, it terminates the request with an error and captures the address and attribute information in the appropriate error registers.

The exact process depends on the target of the request (see Transaction targets).

XRDC optionally supports the inclusion of a hardware semaphore to dynamically alter the ACP of a memory region or peripheral (see Hardware semaphores and dynamic access rights).

18.5.6.2 Transaction targets

The ACP evaluation for an XRDC-protected transaction request depends on the target of the request.

Table 68. Transaction targets

Request target	Topic	Brief description
Peripheral	Peripheral ACP evaluation example	Process for a transaction request to a target peripheral that is within a protected domain, with ACP evaluation configured by PDAC_W*w*_*s*.
Memory	Memory ACP evaluation example	Process for a transaction request to a target memory region that is within a protected domain, with ACP evaluation configured by MRGD_W*w*_*r*.

18.5.6.3 Peripheral ACP evaluation registers

Table 69. Peripheral ACP evaluation registers

Register	Index	Brief description
PDAC_W0_*s*[1]	*s* = peripheral slot	Specifies the ACP for an XRDC-protected peripheral, and an optional hardware semaphore.
PDAC_W1_*s*[1]	*s* = peripheral slot	Enables the set of PDAC registers for the associated peripheral and locks the set. Typically, you configure the peripheral by writing to the PDAC registers and then limiting their respective domains from making updates to the D*d*ACP fields or by locking the set for all updates.

1. See PDAC register structure.

18.5.6.4 PDAC register structure

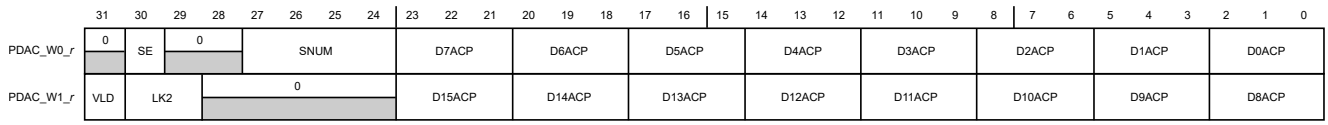


Figure 44. PDAC register structure

18.5.6.5 Memory ACP evaluation registers

When XRDC is enabled (CR[GVLD] = 1), you cannot access any XRDC-protected memory unless you configure at least one set of memory region descriptors (see MRGD_W*w*_*r*) that includes the targeted memory.

Table 70. Memory ACP evaluation registers

Register	Index	Brief description
MRCFG*c*	*c* = memory controller instance	Indicates the number of memory regions per memory controller (NMRGD). Each memory region is configured by a set of memory region descriptor registers (MRGD_W*w*_*r*) described below.
MRGD_W0_*r*[1]	*r* = memory region	Specifies starting address for the memory region.
MRGD_W1_*r*[1]	*r* = memory region	Specifies ending address for the memory region.
MRGD_W2_*r*[1]	*r* = memory region	Specifies the ACP for each supported domain, and an optional hardware semaphore.

Table continues on the next page...

Table 70. Memory ACP evaluation registers (continued)

Register	Index	Brief description
MRGD_W3_*r*[1]	*r* = memory region	Enables the set of MRGD registers for the associated region and locks the set. Typically, you define the memory region by writing to the MRGD registers and then limiting their respective domains from updating the D*d*ACP fields or by locking the set for all updates.

1. See MRGD register structure.

18.5.6.6 MRGD register structure

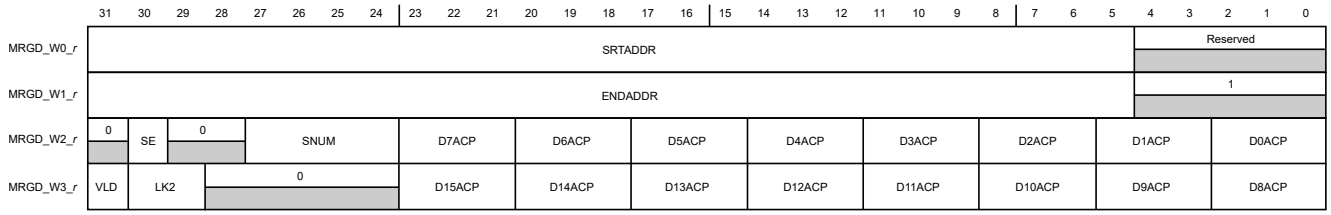


Figure 45. MRGD register structure

18.5.6.7 Access control model

XRDC supports a four-level hierarchical access control model. This model combines the traditional privileged (also known as supervisor) and user access levels with an additional signal for the secure attribute of each memory reference, as shown in Access control model levels.

Each level has different access control policies that specify the read and write accessibility for a target. XRDC combines the privileged and secure attributes with the DID assigned to each system bus transaction to form the hardware basis for the access control mechanism. You specify the ACP for target resources using the D*d*ACP fields (see Domain ACP specification) found in the configuration registers shown in Peripheral ACP evaluation registers and Memory ACP evaluation registers.

You can dynamically control access to shared memory regions and target peripherals with the optional inclusion of a hardware semaphore (see Hardware semaphores and dynamic access rights). If you enable this semaphore for a given address space or peripheral, XRDC allows writes to the target resource only if the requesting domain owns the semaphore.

For cores that support only the three-state access control model (Secure Privileged, Secure User, Nonsecure User), XRDC forces the nonsecure output signal from the MDAC submodule to 0 in privileged mode. This change enables precise state transitions between user and privileged modes. Specifically, the MDAC logic for master *m* generates the nonsecure attribute output signal as a function of the Three-State Model (PID*m*[TSM]) and PID Present (HWCFG2[PIDP*m*]) fields, as shown in Generation of secure attribute).

18.5.6.7.1 Access control model levels

Table 71. Access control model levels

Secure	Nonsecure	Privileged (supervisor)	Not privileged (user)	Level
Yes	Not applicable	Yes	Not applicable	Most restricted
Yes	Not applicable	Not applicable	Yes	More restricted
Not applicable	Yes	Yes	Not applicable	Less restricted
Not applicable	Yes	Not applicable	Yes	Least restricted

18.5.6.7.2 Generation of secure attribute

This table shows how XRDC generates the secure attribute for a transaction.

Table 72. Generation of secure attribute

Access levels [1]	PID*m*[TSM] (b)	PID present [2]	Secure attribute determined by
4	0	No	Master secure attribute
4	0	Yes	Master secure attribute && ~master privileged attribute
3	1	No	PID*m*[5] && ~master privileged attribute
3	1	Yes	Local secure attribute && ~master privileged attribute

1. XRDC assumes a core master supports the four-level access model. If a core supports only the three-state access control model, you must write 1 to PID*m*[TSM] before loading any nonsecure value into the PID.
2. Indicated in HWCFG2–3[PIDP*m*].

18.5.6.8 Domain ACP specification

Table 73. Domain ACP specification

D*d*ACP	Allowable accesses			
	Secure Privileged	Secure User	Nonsecure Privileged	Nonsecure User
111b	R, W	R, W	R, W	R, W
110b	R, W	R, W	R, W	None
101b	R, W	R, W	R	R
100b	R, W	R, W	R	None
011b	R, W	R, W	None	None
010b	R, W	None	None	None
001b	R	R	None	None
000b	None	None	None	None

18.5.6.9 Memory region ACP evaluation

During ACP evaluation of a memory transaction request, if the target memory location falls within the address range specified by any XRDC memory region descriptor, then XRDC identifies the request as a memory region hit. For each memory region hit, XRDC compares the DID, privileged attribute, and secured attribute of the transaction to the associated domain ACP (MRGD_W{2,3}_*r*[D*d*ACP]) in the memory region descriptor. See ACP evaluation for additional details on this function.

The following conditions cause XRDC to report an access error:

- The target memory location does not fall within any defined memory regions; in other words, the transaction request is not a hit.
- The transaction request does not have the appropriate access permissions for the region, which triggers a domain violation.
- The transaction request hits multiple (overlapping) regions, and all of those regions signal access violations.

Unimplemented domain identifiers default to no access privileges, and therefore the access type of a D*d*ACP field for an unimplemented domain is read-as-zero/writes-ignored (RAZ/WI).

18.5.6.10 Hardware semaphores and dynamic access rights

XRDC memory region descriptors and peripheral access control support an optional hardware semaphore in their access evaluations. This hardware semaphore allows hardware enforcement of dynamic access rights, based on the state of the semaphore for shared memory regions or shared peripherals.

If enabled, the state of the semaphore dynamically modifies the access control policies so that only the domain owning it has write access to the resource. The write permissions for all other domains are revoked based on the semaphore state. If no domain owns the semaphore, the PAC and MRC submodules evaluate DdACP normally.

If you enable a hardware semaphore (by writing 1 to MRGD_W2_*r*[SE] or PDAC_W0_*r*[SE]) then, before the normal DdACP evaluation, XRDC checks the state of the hardware semaphore specified in MRGD_W2_*r*[SNUM] or PDAC_W0_*r*[SNUM].

On a write transaction, if the semaphore is not idle (the semaphore state is non-zero) and the requesting domain does not own the semaphore, the memory or peripheral access terminates with an error. In other words, writes into a semaphore-enabled address space or peripheral are allowed only if the semaphore is idle or the requesting domain owns the semaphore.

18.5.7 XRDC transaction examples

To see the complete transaction process for an XRDC-protected transaction request from a master to target:

1. Follow one of these examples of domain assignment:
   - DFMT0 direct domain assignment example
   - DFMT0 PID-based domain assignment example
   - DFMT1 direct domain assignment example-single MDA
2. Then follow one of these examples of ACP evaluation:
   - Peripheral ACP evaluation example
   - Memory ACP evaluation example

18.5.7.1 DFMT0 direct domain assignment example

This configuration assigns a specific domain to all incoming transactions.

- A core master has master number = 6.
- The core has one MDAC register (MDACFG6[NMDAR] = 1), configured as shown in Register settings for DFMT0 direct domain assignment transactions.

MDAC_W*w_m*_DFMT0 registers do not include fields for secure and privileged attributes. Those attributes are part of the transaction data from core masters and are forwarded with the transaction after domain assignment.

18.5.7.1.1 Register settings for DFMT0 direct domain assignment transactions

Table 74. Register settings for DFMT0 direct domain assignment transactions

Field	MDA_W0_6_DFMT0	Comments
VLD	1	Enables this register for use in domain assignment.
LK1	1	Locks the settings in this register until the next module reset.
DFMT	0	Indicates that this is a DFMT0 register.
PID	00_0000b	Not used because PE = 00b.

Table continues on the next page...

Table 74. Register settings for DFMT0 direct domain assignment transactions (continued)

Field	MDA_W0_6_DFMT0	Comments
PIDM	00_0000b	Not used because PE = 00b.
PE	00b	Disables PID-based filtering.
DIDS	00b	All incoming transactions are to be assigned to the domain specified by the DID field.
DID	01b	Because PE = 00b and DIDS = 00b, all incoming transactions are to be assigned DID value 01b.

18.5.7.1.2 DFMT0 direct domain assignment process

The application configures XRDC as it boots. After application boot completes, a core issues a read request to a target peripheral.

1. XRDC intercepts the request and performs domain assignment using the configuration in MDA_W0_6_DFMT0. In this example, XRDC assigns DID = 01b to all incoming transactions.
2. The transaction proceeds with DID = 01b and the privileged and secure attributes provided by the core.

18.5.7.2 DFMT0 PID-based domain assignment example

This example configuration demonstrates PID-based domain assignment:

- A core master with master number = 4 processes both safety-critical tasks and routine tasks, using two domains:
  - Domain 0 is reserved for safety-critical tasks (PID = 0–15).
  - Domain 1 is reserved for routine tasks (PID > 15).
- The core has eight MDAC registers (MDACFG4[NMDAR] = 8), configured as follows (see Register settings for DFMT0 PID-based transactions):
  - MDA_W0_4_DFMT0 assigns safety-critical tasks (PID = 0–15) to domain 0.
  - MDA_W1_4_DFMT0 assigns routine tasks (PID = 16–31) to domain 1.
  - MDA_W2_4_DFMT0 assigns routine tasks (PID = 32–63) to domain 1.
  - MDA_W3_4_DFMT0 through MDA_W7_4_DFMT0 are not used.

MDAC_W*w_m*_DFMT0 registers do not include fields for secure and privileged attributes. Those attributes are part of the transaction data from core masters and are forwarded with the transaction after domain assignment.

18.5.7.2.1 Register settings for DFMT0 PID-based transactions

Table 75. Register settings for PID-based transactions

Field	Registers				
	MDA_W0_4_DFMT0	MDA_W1_4_DFMT0	MDA_W2_4_DFMT0	MDA_W[3–7]_4_DFMT0	Comments
VLD	1	1	1	0	Enables this register for use when assigning domains.
LK1	1	1	1	1	Locks the settings in each register until the next device reset.
DFMT	0	0	0	0	Indicates that these registers apply to domain assignment for core masters.

Table continues on the next page...

Table 75. Register settings for PID-based transactions (continued)

Field	Registers				
	MDA_W0_4_DFMT0	MDA_W1_4_DFMT0	MDA_W2_4_DFMT0	MDA_W[3–7]_4_DFMT0	Comments
PID	00_0000b	01_0000b	10_0000b	—	Constant match value to be used for PID-based filtering.
PIDM	00_1111b	10_1111b	01_1111b	—	Each 0 bit causes the corresponding PID bit to be considered in domain assignment.
PE	10b	10b	10b	—	Specifies the type of pattern matching used for PID evaluation.
DIDS	00b	00b	00b	—	Assign all incoming transactions with PIDs that pass the filtering criteria to the domain specified by the DID field.
DID	00b	01b	01b	—	This DID value is assigned to incoming transactions with PIDs that pass the filtering criteria.

18.5.7.2.2 DFMT0 PID-based domain assignment process

The application configures XRDC as it boots. After booting completes, a core issues a read request to a target peripheral. The task making the request has PID = 6, indicating that it is a safety-critical task.

1. XRDC intercepts the request and performs domain assignment using each enabled MDA_W*n*_4_DFMT0 register, regardless of whether it has already found a match. In this example, XRDC performs the domain assignments as shown in DFMT0 PID-based domain assignment evaluation.
2. After XRDC completes all domain assignment evaluations, it logically ORs the assigned DIDs to determine the final DID assigned to the transaction. In this example, only one evaluation results in a DID assignment, so there is no OR operation.
3. The transaction proceeds with DID 00b and the privilege and secure attributes provided by the core.

18.5.7.2.3 DFMT0 PID-based domain assignment evaluation

Table 76. DFMT0 PID-based domain assignment evaluation

Register	Evaluation steps	Boolean math	Result
MDA_W0_4_DFMT0	1. Bitwise AND of PID with inverted PIDM.	00_0000b & 11_0000b	00_0000b
	2. Bitwise AND of transaction PID (PID4[PID]) with inverted PIDM.	000110b & 11_0000b	00_0000b
	3. Compare the results of steps 1 and 2.	00_0000b == 000000b	True: Assign DID 0
MDA_W1_4_DFMT0	1. Bitwise AND of PID with inverted PIDM.	01_0000b & 01_0000b	01_0000b
	2. Bitwise AND of transaction PID (PID4[PID]) with inverted PIDM.	00_0110b & 01_0000b	00_0000b
	3. Compare the results of steps 1 and 2.	01_0000b == 00_0000b	False: No DID assignment
MDA_W2_4_DFMT0	1. Bitwise AND of PID with inverted PIDM.	10_0000b & 10_0000b	10_0000b

Table continues on the next page...

Table 76. DFMT0 PID-based domain assignment evaluation (continued)

Register	Evaluation steps	Boolean math	Result
	2. Bitwise AND of transaction PID (PID4[PID]) with inverted PIDM.	00_0110b & 10_0000b	00_0000b
	3. Compare the results of steps 1 and 2.	10_0000b == 00_0000b	False: No DID assignment

18.5.7.3 DFMT1 direct domain assignment example-single MDA

This configuration assigns a specific domain to all incoming transactions.

- A master has master number = 6.
- The master has one MDAC register (MDACFG6[NMDAR] = 1) as shown in Register settings for DFMT1 direct domain assignment transactions.

18.5.7.3.1 Register settings for DFMT1 direct domain assignment transactions

Table 77. Register settings for DFMT1 direct domain assignment transactions

Field	MDA_W0_6_DFMT1	Comments
VLD	1	Enables this register for use in domain assignment.
LK1	1	Locks the settings in this register until the next chip reset.
DFMT	1	Indicates that this is a DFMT1 register.
DIDB	0b	All incoming transactions are to be assigned to the domain that the DID field specifies.
SA	10b	All incoming transactions retain their Secure attribute value.
PA	10b	All incoming transactions retain their Privileged attribute value.
DID	01b	Because DIDB = 0b, all incoming transactions are to be assigned DID value 01b.

18.5.7.3.2 DFMT1 direct domain assignment process

The application configures XRDC as it boots. After application boot completes, a master issues a read request to a target peripheral.

1. XRDC intercepts the request and assigns a domain using the configuration in MDA_W0_6_DFMT1. In this example, XRDC assigns DID = 01b to all incoming transactions.
2. The transaction proceeds with DID = 01b and the privileged and secure attributes provided by the master.

18.5.7.4 Peripheral ACP evaluation example

This example configuration demonstrates ACP evaluation for a target peripheral.

- The core master must have highly available, exclusive access to the ADC0 peripheral for safety-critical tasks in domain 0.
- The core master supports 8 domains (HWCFG0[NDID] = 111b).
- ADC occupies PDAC slot 40 in the chip, as defined in the memory map file attached to this document (see Finding the PDAC slot number for a peripheral).
- Given 8 domains and PDAC slot 40, the PDAC registers associated with ADC0 are PDAC_W0_40 and PDAC_W1_40.

The following sections describe the register configurations for this example.

18.5.7.4.1 Finding the PDAC slot number for a peripheral

This topic shows how to find the PDAC slot number for a peripheral, but it is a generic example. Memory map file organization and appearance can vary.

To find the PDAC slot number for a peripheral:

1. Open the memory map file attached to this document and view the peripherals page.
2. Locate the peripheral in the "Instance" column.
3. The PDAC slot number for the peripheral is at the intersection of the "PDAC slot number" column and the peripheral row.

In this figure, for peripheral ADC_0, the PDAC slot number is 40. Therefore, the PDAC registers for ADC0 are PDAC_W*w*_40.

	A	B	D	E	I	J
1	Instance	Description	Start address	End address	PDAC slot number	AIPS instance
8	LCU_0	Logic Control Unit 0	0x40098000	0x4009BFFF	38	AIPS_0
9	LCU_1	Logic Control Unit 1	0x4009C000	0x4009FFFF	39	AIPS_0
10	ADC_0	Analog-to-digital converter 0	0x400A0000	0x400A3FFF	40	AIPS_0
11	ADC_1	Analog-to-digital converter 1	0x400A4000	0x400A7FFF	41	AIPS_0
12	ADC_2	Analog-to-digital converter 2	0x400A8000	0x400ABFFF	42	AIPS_0

Figure 46. Finding the PDAC slot number for a peripheral

18.5.7.4.2 Register settings for peripheral ACP evaluation

This table shows the PDAC_W*w*_40 settings for a safety-critical task assigned to domain 0.

Table 78. Register settings for peripheral ACP evaluation

Register	Field	Value (b)	Comments
PDAC_W0_40	SE	0	The hardware semaphore (see the SEMA42 chapter) is disabled.
	SNUM	(don't care)	The hardware semaphore is not used in this example.
	D7ACP	000	Domain 7 has no access to the peripheral.
	D6ACP	000	Domain 6 has no access to the peripheral.
	D5ACP	000	Domain 5 has no access to the peripheral.
	D4ACP	000	Domain 4 has no access to the peripheral.
	D3ACP	000	Domain 3 has no access to the peripheral.
	D2ACP	000	Domain 2 has no access to the peripheral.
	D1ACP	000	Domain 1 has no access to the peripheral.
	D0ACP	010	Only privileged, secure transactions from domain 0 have access.
PDAC_W1_40	VLD	1	Use this register set in domain ACP evaluations.
	LK2	11	Lock the settings in this register until the next device reset.

18.5.7.4.3 Peripheral ACP evaluation process

XRDC performs the following process for ACP evaluation:

1. When the application is running, the core issues a read request to a target peripheral. The task making the request has PID = 0, indicating that it is a safety-critical task. The transaction request is privileged and secured.
2. Between the chip interconnect and ADC0, XRDC intercepts the request and compares its DID, privileged attribute, and secured attribute to the configuration in PDAC_W0_40 and PDAC_W1_40.
3. Because the ADC0 D0ACP field is 010b for privileged, secured access, XRDC grants access to the transaction.
4. The transaction proceeds normally without any further intervention from XRDC.

18.5.7.5 Memory ACP evaluation example

This example configuration demonstrates ACP evaluation for a target memory. Following are the desired features:

- The core master must have highly available, exclusive access to the memory region for safety-critical tasks in domain 0.
- The target memory is the address range 1B00_0000h–1B00_1FFFh, protected by memory controller 0 (MRC0).
- Access to the entire memory range will be controlled by the memory region descriptor defined by MRGD_W*w*_0.
- The requested transaction is secure privileged.

With the configuration settings shown in Register settings for memory ACP evaluation, XRDC grants access and the transaction proceeds normally. There is no further XRDC intervention.

18.5.7.5.1 Register settings for memory ACP evaluation

Table 79. Register settings for memory ACP evaluation

Register	Field	Value	Comments
MRGD_W0_0	SRTADDR	1B00_0000h	Starting address of the memory region.
MRGD_W1_0	ENDADDR	1B00_1FFFh	Ending address of the memory region.
MRGD_W2_0	SE	0	The hardware semaphore (see the SEMA42 chapter) is disabled.
	SNUM	(don't care)	The hardware semaphore is not used in this example.
	D7ACP	000b	Domain 7 has no access to the peripheral.
	D6ACP	000b	Domain 6 has no access to the peripheral.
	D5ACP	000b	Domain 5 has no access to the peripheral.
	D4ACP	000b	Domain 4 has no access to the peripheral.
	D3ACP	000b	Domain 3 has no access to the peripheral.
	D2ACP	000b	Domain 2 has no access to the peripheral.
	D1ACP	000b	Domain 1 has no access to the peripheral.
	D0ACP	010b	Only privileged, secure transactions from domain 0 have access.
MRGD_W3_0	VLD	1	Use this register set in domain ACP evaluations.
	LK2	11b	Lock the settings in this register until the next device reset.

18.5.7.5.2 Memory ACP evaluation process

In this example, XRDC performs the following process for ACP evaluation:

1. When the application is running, the core issues a read request to address 1B00_0100h. The transaction request is secure privileged.

2. Between the chip interconnect and the memory, XRDC intercepts the request and compares its DID, memory location of the address, privileged attribute, and secured attribute to the configuration in MRGD_W0_0, MRGD_W1_0, MRGD_W2_0 and MRGD_W3_0.
3. Because the address 1B00_0100h is in the memory range 1B00_0000h–1B00_1FFFh and its D0ACP field is 010b for privileged, secured access, XRDC grants access to the transaction.
4. The transaction proceeds normally without any no further intervention from XRDC.

18.5.8 Clocking

This module has no clocking considerations.

18.5.9 Interrupts

This module outputs an interrupt signal which can be connected to interrupt controller. Check chip-specific interrupt assignment for details. Interrupt is asserted on detection of access violation by any checker, and it remains asserted until DERRLOC registers are cleared.

18.6 Initialization information

Out of reset, XRDC is disabled (CR[GVLD] = 0), which allows secure privileged startup code to configure the entire programming model.

18.6.1 Initialization procedure

1. Read the hardware configuration registers to obtain the implemented XRDC hardware capabilities for the chip:
   - HWCFG0
   - HWCFG1
   - HWCFG2
   - MDACFG*m* (one for each supported bus master—number indicated in HWCFG0[NMSTR])
   - MRCFG*c* (one for each supported memory controller—number indicated in HWCFG0[NMRC])
2. Use the information retrieved in step 1 and the desired domain architecture to configure:
   - Domain assignments (MDA_W*w_m*_DFMT0 and MDA_W*w_m*_DFMT1)
   - Memory region descriptors (MRGD_W*w_n*)
   - Peripheral access control (PDAC_W*w_s*)

   Ensure that you enable the necessary registers and register sets using the appropriate VLD fields. Also, you can limit access to these registers or lock them after you configure them, by using the appropriate LK1 fields.
3. Enable XRDC (write 1 to CR[GVLD]).

XRDC is now fully operational.

18.6.2 Minimize access errors

When you configure and enable XRDC, it begins generating DIDs for transaction requests and evaluating access rights at the target memory and peripheral resources. Because of the distributed design hierarchy and the pipelined nature of the hardware system bus fabric, it can take multiple cycles for a generated DID to propagate. Until that happens, XRDC uses the master's default DID. Depending on the programmed ACPs, the default DID might generate an access error response.

If XRDC generates incorrect error responses, you can use the following approaches to minimize or eliminate these extraneous access errors:

- Minimize the amount of system bus traffic when XRDC is enabled (CR[GVLD] = 1).

- Ensure that all target memory addresses provide sufficient access rights for any default DIDs and for the newly programmed DIDs. After XRDC is fully operational, as confirmed by a read of HWCFG1, you can remove the permissions for the default DIDs.
- Try to have the bus master that programs and configures XRDC use the same DID, that is, its default DID, for both initialization and configuration, besides normal system operation. You do this when you define the DID assignments for the system.

18.7 Application information

18.7.1 Master domain assignments

The typical use case for master domain assignments is to include one or more core bus masters in a single domain, possibly combined with other noncore bus master modules such as DMA. This configuration may be static or may be changed dynamically to select between a small number of domains. HWCFG0[NDID] indicates the maximum number of supported domains. XRDC also supports the optional use of PIDs to create multiple classes of cores, each in different domains.

For example, you can group critical tasks—safety-critical, performance-critical, and so on—into one domain and all other tasks into a second domain. Typically, you assign the DID at initialization, but you can also reconfigure domain assignment while the application is running.

A core bus master typically has multiple MDA_W*w_m*_DFMT0 registers associated with it.

A noncore bus master typically has a single MDA_W*w_m*_DFMT1 register associated with it.

The master domain assignment, memory region descriptor, and peripheral domain access control registers have lock fields that enable you to limit access to, or to lock, the registers. These actions protect the configuration.

18.7.2 Memory region descriptor management

There are two important concepts to consider for managing the memory region descriptors.

Each MRC*c* configuration is chip-specific. See the chip-specific XRDC information for the number of implemented memory region descriptors (MRGD_W*w_n*) in a given MRC*c* instance, and the specific port numbers associated with the target memories being monitored.

Second, as detailed in Memory region ACP evaluation, after you enable the XRDC, a memory reference must hit one or more of the configured regions. Otherwise, the transaction results in an access violation. Two other conditions also result in access errors:

- The access hits a single region descriptor and that region signals a domain violation.
- The access hits multiple (overlapping) regions and all regions signal violations.

The second condition reflects that XRDC gives priority to permission granting over access denying for overlapping regions. This approach provides more flexibility to system software in memory region descriptor assignments.

18.7.3 Domain error capture management

18.7.3.1 Domain error capture registers

When an MRC or PAC detects a domain access violation, XRDC captures information about the transaction in the following registers:

Table 80. Domain error capture registers

Register[field]	Index	Information
DERRLOC*d*[MRCINST]	*d* = DID	Domain error location for MRC instances, with asserted bits indicating which MRC instance numbers are reporting an error

Table continues on the next page...

Table 80. Domain error capture registers (continued)

Register[field]	Index	Information
DERRLOC*d*[PACINST]	*d* = DID	Domain error location for PAC instances, with asserted bits indicating which PAC instance numbers are reporting an error
DERR_W0_*i*	*i* = instance number	Transaction target address
DERR_W1_*i*	*i* = instance number	Additional information about the transaction
DERR_W3_*i*	*i* = instance number	Reset and rearm domain error capture for the instance

18.7.3.2 Handling domain access violation errors

When an MRC or PAC instance detects a domain access violation, it reports the error by asserting the associated bit in DERRLOCd[MRCINST] or [PACINST], and XRDC asserts the error interrupt output. To retrieve information about the error, the error handler must:

1. Read each DERRLOC*d* register until it finds a non-zero MRCINST or PACINST value.

   The index of the DERRLOC*d* register is the DID for the domain in which the error occurred.

2. Configure the domain assignment for the master executing the exception handler (MDA_Ww_m_DFMTf[DID]) to assign the DID that corresponds to the DERRLOC*d* register index.

3. Read HWCFG1[DID] to be sure the error handler is now operating in the correct domain. In other words, make sure HWCFG1[DID] equals the value written to MDA_Ww_m_DFMTf[DID].

4. Find the number of an MRC or PAC instance reporting an error by parsing DERRLOC*d*[MRCINST] and DERRLOC*d*[PACINST] for an asserted bit.

   There may be multiple access violations, across multiple MRC or PAC instances, pending for a given domain. To quickly find the lowest numbered instance reporting an access violation, execute a "find first one bit" instruction (alternatively known as "count leading zeroes") on the MRCINST and PACINST fields.

5. Retrieve the error address (DERR_W0_*i*[EADDR]).

6. Retrieve the error information (DERR_W1_*i*).

   More than one error may have occurred in the MRC or PAC instance, as indicated by the error status (DERR_W1_*i*[EST] = 11b). If more than one error has occurred in the instance, XRDC captures data only for the first error.

7. Use the error address and information to handle the error (whatever that may require).

8. Reset the DERR_W*w_i* registers and rearm error capture (write 1b to DERR_W3_*i*[RECR]).

   Rearming error capture deasserts the instance bit in DERRLOC*d*.

9. Repeat steps 1 and 8 for each asserted bit in PACINST and MRCINST until there are no more asserted bits.

Domain error retrieval illustrates an example error retrieval.

18.7.3.3 Domain error retrieval

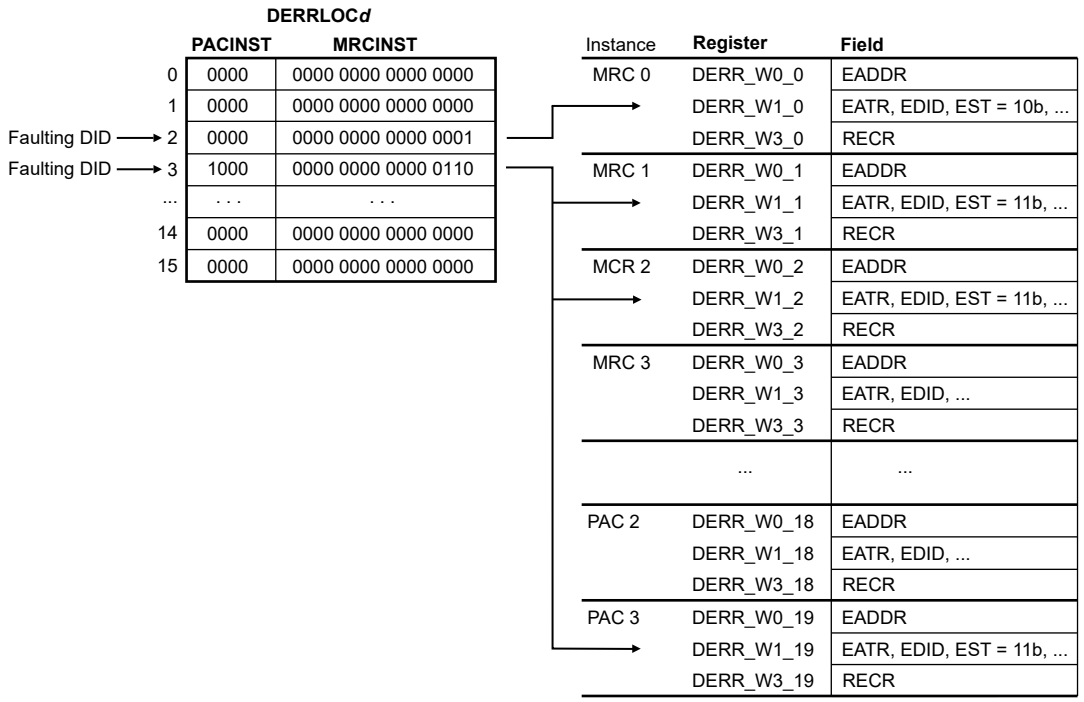


Figure 47. Domain error retrieval

18.8 Memory map and register definitions

18.8.1 Register organization

XRDC registers are partitioned into these groups:

- Basic hardware control and configuration
- Domain errors (including location and details)
- Master domain assignments
- Peripheral domain access controls
- Memory region descriptors

18.8.2 Register access guidelines

The following guidelines apply to XRDC register access:

- You can access the XRDC registers only in secure, privileged access mode.
- Unless stated otherwise, the registers support 8-, 16-, and 32-bit reads, and 32-bit writes.
- Unless stated otherwise, XRDC terminates the following access attempts with an error:
  - Accesses in a different access mode
  - Unsupported write data size
  - Writes to read-only resources
  - Writes to reserved address spaces

- Accesses to these memory map holes return an error:
  - Any access to a register that does not exist
  - Holes in the 0–F0h and DERR to PID register space
  - For MDAC, gaps in the master domain assignment (MDA_W*w*_*m*_DFMT*n*) registers
  - For MRCs, any gap in the MRGD_W*w*_*r* registers (for example, if there are four memory region descriptors, attempted access to a fifth descriptor fails)
- Accesses to these memory map holes do not return a bus error:
  - MRCs: Memory region descriptors occupy only four words but have an additional four words of address available: words 4–7
  - PDAC: Registers associated with unimplemented PDAC slots
- Read accesses to these memory map holes do not return a bus error:
  - Offset F8h and FCh
  - Offset 100–13Fh
  - Offset 140–14Fh
  - Offset 200–23Ch

18.8.3 XRDC register descriptions

18.8.3.1 XRDC memory map

XRDC base address: 4027_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	0000_008Ah
F0h	Hardware Configuration 0 (HWCFG0)	32	R	1202_0502h
F4h	Hardware Configuration 1 (HWCFG1)	32	R	See section
F8h	Hardware Configuration 2 (HWCFG2)	32	R	0000_0000h
100h	Master Domain Assignment Configuration (MDACFG0)	8	R	01h
101h	Master Domain Assignment Configuration (MDACFG1)	8	R	81h
102h	Master Domain Assignment Configuration (MDACFG2)	8	R	81h
103h	Master Domain Assignment Configuration (MDACFG3)	8	R	01h
104h	Master Domain Assignment Configuration (MDACFG4)	8	R	01h
105h	Master Domain Assignment Configuration (MDACFG5)	8	R	81h
140h	Memory Region Configuration (MRCFG0)	8	R	10h
141h	Memory Region Configuration (MRCFG1)	8	R	10h
142h	Memory Region Configuration (MRCFG2)	8	R	04h
200h - 208h	Domain Error Location (DERRLOC0 - DERRLOC2)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
400h	Domain Error Word 0 (DERR_W0_0)	32	R	0000_0000h
404h	Domain Error Word 1 (DERR_W1_0)	32	R	0000_0000h
40Ch	Domain Error Word 3 (DERR_W3_0)	32	RW	0000_0000h
410h	Domain Error Word 0 (DERR_W0_1)	32	R	0000_0000h
414h	Domain Error Word 1 (DERR_W1_1)	32	R	0000_0000h
41Ch	Domain Error Word 3 (DERR_W3_1)	32	RW	0000_0000h
420h	Domain Error Word 0 (DERR_W0_2)	32	R	0000_0000h
424h	Domain Error Word 1 (DERR_W1_2)	32	R	0000_0000h
42Ch	Domain Error Word 3 (DERR_W3_2)	32	RW	0000_0000h
500h	Domain Error Word 0 (DERR_W0_16)	32	R	0000_0000h
504h	Domain Error Word 1 (DERR_W1_16)	32	R	0000_0000h
50Ch	Domain Error Word 3 (DERR_W3_16)	32	RW	0000_0000h
510h	Domain Error Word 0 (DERR_W0_17)	32	R	0000_0000h
514h	Domain Error Word 1 (DERR_W1_17)	32	R	0000_0000h
51Ch	Domain Error Word 3 (DERR_W3_17)	32	RW	0000_0000h
520h	Domain Error Word 0 (DERR_W0_18)	32	R	0000_0000h
524h	Domain Error Word 1 (DERR_W1_18)	32	R	0000_0000h
52Ch	Domain Error Word 3 (DERR_W3_18)	32	RW	0000_0000h
700h	Process Identifier (PID0)	32	RW	0000_0000h
70Ch	Process Identifier (PID3)	32	RW	0000_0000h
710h	Process Identifier (PID4)	32	RW	0000_0000h
800h	Master Domain Assignment (MDA_W0_0_DFMT0)	32	RW	0000_0000h
820h	Master Domain Assignment (MDA_W0_1_DFMT1)	32	RW	2000_0000h
840h	Master Domain Assignment (MDA_W0_2_DFMT1)	32	RW	2000_0000h
860h	Master Domain Assignment (MDA_W0_3_DFMT0)	32	RW	0000_0000h
880h	Master Domain Assignment (MDA_W0_4_DFMT0)	32	RW	0000_0000h
8A0h	Master Domain Assignment (MDA_W0_5_DFMT1)	32	RW	2000_0000h
1100h	Peripheral Domain Access Control Word 0 (PDAC_W0_32)	32	RW	0000_0000h
1104h	Peripheral Domain Access Control Word 1 (PDAC_W1_32)	32	RW	0000_0000h
1108h	Peripheral Domain Access Control Word 0 (PDAC_W0_33)	32	RW	0000_0000h
110Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_33)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1110h	Peripheral Domain Access Control Word 0 (PDAC_W0_34)	32	RW	0000_0000h
1114h	Peripheral Domain Access Control Word 1 (PDAC_W1_34)	32	RW	0000_0000h
1118h	Peripheral Domain Access Control Word 0 (PDAC_W0_35)	32	RW	0000_0000h
111Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_35)	32	RW	0000_0000h
1120h	Peripheral Domain Access Control Word 0 (PDAC_W0_36)	32	RW	0000_0000h
1124h	Peripheral Domain Access Control Word 1 (PDAC_W1_36)	32	RW	0000_0000h
1130h	Peripheral Domain Access Control Word 0 (PDAC_W0_38)	32	RW	0000_0000h
1134h	Peripheral Domain Access Control Word 1 (PDAC_W1_38)	32	RW	0000_0000h
1138h	Peripheral Domain Access Control Word 0 (PDAC_W0_39)	32	RW	0000_0000h
113Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_39)	32	RW	0000_0000h
1140h	Peripheral Domain Access Control Word 0 (PDAC_W0_40)	32	RW	0000_0000h
1144h	Peripheral Domain Access Control Word 1 (PDAC_W1_40)	32	RW	0000_0000h
1148h	Peripheral Domain Access Control Word 0 (PDAC_W0_41)	32	RW	0000_0000h
114Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_41)	32	RW	0000_0000h
1150h	Peripheral Domain Access Control Word 0 (PDAC_W0_42)	32	RW	0000_0000h
1154h	Peripheral Domain Access Control Word 1 (PDAC_W1_42)	32	RW	0000_0000h
1160h	Peripheral Domain Access Control Word 0 (PDAC_W0_44)	32	RW	0000_0000h
1164h	Peripheral Domain Access Control Word 1 (PDAC_W1_44)	32	RW	0000_0000h
1168h	Peripheral Domain Access Control Word 0 (PDAC_W0_45)	32	RW	0000_0000h
116Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_45)	32	RW	0000_0000h
1170h	Peripheral Domain Access Control Word 0 (PDAC_W0_46)	32	RW	0000_0000h
1174h	Peripheral Domain Access Control Word 1 (PDAC_W1_46)	32	RW	0000_0000h
1178h	Peripheral Domain Access Control Word 0 (PDAC_W0_47)	32	RW	0000_0000h
117Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_47)	32	RW	0000_0000h
1400h	Peripheral Domain Access Control Word 0 (PDAC_W0_128)	32	RW	0000_0000h
1404h	Peripheral Domain Access Control Word 1 (PDAC_W1_128)	32	RW	0000_0000h
1408h	Peripheral Domain Access Control Word 0 (PDAC_W0_129)	32	RW	0000_0000h
140Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_129)	32	RW	0000_0000h
1410h	Peripheral Domain Access Control Word 0 (PDAC_W0_130)	32	RW	0000_0000h
1414h	Peripheral Domain Access Control Word 1 (PDAC_W1_130)	32	RW	0000_0000h
1418h	Peripheral Domain Access Control Word 0 (PDAC_W0_131)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
141Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_131)	32	RW	0000_0000h
1420h	Peripheral Domain Access Control Word 0 (PDAC_W0_132)	32	RW	0000_0000h
1424h	Peripheral Domain Access Control Word 1 (PDAC_W1_132)	32	RW	0000_0000h
1428h	Peripheral Domain Access Control Word 0 (PDAC_W0_133)	32	RW	0000_0000h
142Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_133)	32	RW	0000_0000h
1430h	Peripheral Domain Access Control Word 0 (PDAC_W0_134)	32	RW	0000_0000h
1434h	Peripheral Domain Access Control Word 1 (PDAC_W1_134)	32	RW	0000_0000h
1438h	Peripheral Domain Access Control Word 0 (PDAC_W0_135)	32	RW	0000_0000h
143Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_135)	32	RW	0000_0000h
1440h	Peripheral Domain Access Control Word 0 (PDAC_W0_136)	32	RW	0000_0000h
1444h	Peripheral Domain Access Control Word 1 (PDAC_W1_136)	32	RW	0000_0000h
1448h	Peripheral Domain Access Control Word 0 (PDAC_W0_137)	32	RW	0000_0000h
144Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_137)	32	RW	0000_0000h
1450h	Peripheral Domain Access Control Word 0 (PDAC_W0_138)	32	RW	0000_0000h
1454h	Peripheral Domain Access Control Word 1 (PDAC_W1_138)	32	RW	0000_0000h
1458h	Peripheral Domain Access Control Word 0 (PDAC_W0_139)	32	RW	0000_0000h
145Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_139)	32	RW	0000_0000h
1460h	Peripheral Domain Access Control Word 0 (PDAC_W0_140)	32	RW	0000_0000h
1464h	Peripheral Domain Access Control Word 1 (PDAC_W1_140)	32	RW	0000_0000h
1468h	Peripheral Domain Access Control Word 0 (PDAC_W0_141)	32	RW	0000_0000h
146Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_141)	32	RW	0000_0000h
1470h	Peripheral Domain Access Control Word 0 (PDAC_W0_142)	32	RW	0000_0000h
1474h	Peripheral Domain Access Control Word 1 (PDAC_W1_142)	32	RW	0000_0000h
1478h	Peripheral Domain Access Control Word 0 (PDAC_W0_143)	32	RW	0000_0000h
147Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_143)	32	RW	0000_0000h
1480h	Peripheral Domain Access Control Word 0 (PDAC_W0_144)	32	RW	0000_0000h
1484h	Peripheral Domain Access Control Word 1 (PDAC_W1_144)	32	RW	0000_0000h
1488h	Peripheral Domain Access Control Word 0 (PDAC_W0_145)	32	RW	0000_0000h
148Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_145)	32	RW	0000_0000h
1490h	Peripheral Domain Access Control Word 0 (PDAC_W0_146)	32	RW	0000_0000h
1494h	Peripheral Domain Access Control Word 1 (PDAC_W1_146)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1498h	Peripheral Domain Access Control Word 0 (PDAC_W0_147)	32	RW	0000_0000h
149Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_147)	32	RW	0000_0000h
14A0h	Peripheral Domain Access Control Word 0 (PDAC_W0_148)	32	RW	0000_0000h
14A4h	Peripheral Domain Access Control Word 1 (PDAC_W1_148)	32	RW	0000_0000h
14A8h	Peripheral Domain Access Control Word 0 (PDAC_W0_149)	32	RW	0000_0000h
14ACh	Peripheral Domain Access Control Word 1 (PDAC_W1_149)	32	RW	0000_0000h
14B0h	Peripheral Domain Access Control Word 0 (PDAC_W0_150)	32	RW	0000_0000h
14B4h	Peripheral Domain Access Control Word 1 (PDAC_W1_150)	32	RW	0000_0000h
14B8h	Peripheral Domain Access Control Word 0 (PDAC_W0_151)	32	RW	0000_0000h
14BCh	Peripheral Domain Access Control Word 1 (PDAC_W1_151)	32	RW	0000_0000h
14C0h	Peripheral Domain Access Control Word 0 (PDAC_W0_152)	32	RW	0000_0000h
14C4h	Peripheral Domain Access Control Word 1 (PDAC_W1_152)	32	RW	0000_0000h
14C8h	Peripheral Domain Access Control Word 0 (PDAC_W0_153)	32	RW	0000_0000h
14CCh	Peripheral Domain Access Control Word 1 (PDAC_W1_153)	32	RW	0000_0000h
14D0h	Peripheral Domain Access Control Word 0 (PDAC_W0_154)	32	RW	0000_0000h
14D4h	Peripheral Domain Access Control Word 1 (PDAC_W1_154)	32	RW	0000_0000h
14D8h	Peripheral Domain Access Control Word 0 (PDAC_W0_155)	32	RW	0000_0000h
14DCh	Peripheral Domain Access Control Word 1 (PDAC_W1_155)	32	RW	0000_0000h
14E0h	Peripheral Domain Access Control Word 0 (PDAC_W0_156)	32	RW	0000_0000h
14E4h	Peripheral Domain Access Control Word 1 (PDAC_W1_156)	32	RW	0000_0000h
14E8h	Peripheral Domain Access Control Word 0 (PDAC_W0_157)	32	RW	0000_0000h
14ECh	Peripheral Domain Access Control Word 1 (PDAC_W1_157)	32	RW	0000_0000h
14F0h	Peripheral Domain Access Control Word 0 (PDAC_W0_158)	32	RW	0000_0000h
14F4h	Peripheral Domain Access Control Word 1 (PDAC_W1_158)	32	RW	0000_0000h
14F8h	Peripheral Domain Access Control Word 0 (PDAC_W0_159)	32	RW	0000_0000h
14FCh	Peripheral Domain Access Control Word 1 (PDAC_W1_159)	32	RW	0000_0000h
1500h	Peripheral Domain Access Control Word 0 (PDAC_W0_160)	32	RW	0000_0000h
1504h	Peripheral Domain Access Control Word 1 (PDAC_W1_160)	32	RW	0000_0000h
1508h	Peripheral Domain Access Control Word 0 (PDAC_W0_161)	32	RW	0000_0000h
150Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_161)	32	RW	0000_0000h
1510h	Peripheral Domain Access Control Word 0 (PDAC_W0_162)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1514h	Peripheral Domain Access Control Word 1 (PDAC_W1_162)	32	RW	0000_0000h
1518h	Peripheral Domain Access Control Word 0 (PDAC_W0_163)	32	RW	0000_0000h
151Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_163)	32	RW	0000_0000h
1520h	Peripheral Domain Access Control Word 0 (PDAC_W0_164)	32	RW	0000_0000h
1524h	Peripheral Domain Access Control Word 1 (PDAC_W1_164)	32	RW	0000_0000h
1528h	Peripheral Domain Access Control Word 0 (PDAC_W0_165)	32	RW	0000_0000h
152Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_165)	32	RW	0000_0000h
1530h	Peripheral Domain Access Control Word 0 (PDAC_W0_166)	32	RW	0000_0000h
1534h	Peripheral Domain Access Control Word 1 (PDAC_W1_166)	32	RW	0000_0000h
1538h	Peripheral Domain Access Control Word 0 (PDAC_W0_167)	32	RW	0000_0000h
153Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_167)	32	RW	0000_0000h
1540h	Peripheral Domain Access Control Word 0 (PDAC_W0_168)	32	RW	0000_0000h
1544h	Peripheral Domain Access Control Word 1 (PDAC_W1_168)	32	RW	0000_0000h
1548h	Peripheral Domain Access Control Word 0 (PDAC_W0_169)	32	RW	0000_0000h
154Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_169)	32	RW	0000_0000h
1550h	Peripheral Domain Access Control Word 0 (PDAC_W0_170)	32	RW	0000_0000h
1554h	Peripheral Domain Access Control Word 1 (PDAC_W1_170)	32	RW	0000_0000h
1558h	Peripheral Domain Access Control Word 0 (PDAC_W0_171)	32	RW	0000_0000h
155Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_171)	32	RW	0000_0000h
1568h	Peripheral Domain Access Control Word 0 (PDAC_W0_173)	32	RW	0000_0000h
156Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_173)	32	RW	0000_0000h
1578h	Peripheral Domain Access Control Word 0 (PDAC_W0_175)	32	RW	0000_0000h
157Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_175)	32	RW	0000_0000h
1588h	Peripheral Domain Access Control Word 0 (PDAC_W0_177)	32	RW	0000_0000h
158Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_177)	32	RW	0000_0000h
1590h	Peripheral Domain Access Control Word 0 (PDAC_W0_178)	32	RW	0000_0000h
1594h	Peripheral Domain Access Control Word 1 (PDAC_W1_178)	32	RW	0000_0000h
1598h	Peripheral Domain Access Control Word 0 (PDAC_W0_179)	32	RW	0000_0000h
159Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_179)	32	RW	0000_0000h
15A0h	Peripheral Domain Access Control Word 0 (PDAC_W0_180)	32	RW	0000_0000h
15A4h	Peripheral Domain Access Control Word 1 (PDAC_W1_180)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
15A8h	Peripheral Domain Access Control Word 0 (PDAC_W0_181)	32	RW	0000_0000h
15ACh	Peripheral Domain Access Control Word 1 (PDAC_W1_181)	32	RW	0000_0000h
15B0h	Peripheral Domain Access Control Word 0 (PDAC_W0_182)	32	RW	0000_0000h
15B4h	Peripheral Domain Access Control Word 1 (PDAC_W1_182)	32	RW	0000_0000h
15B8h	Peripheral Domain Access Control Word 0 (PDAC_W0_183)	32	RW	0000_0000h
15BCh	Peripheral Domain Access Control Word 1 (PDAC_W1_183)	32	RW	0000_0000h
15C0h	Peripheral Domain Access Control Word 0 (PDAC_W0_184)	32	RW	0000_0000h
15C4h	Peripheral Domain Access Control Word 1 (PDAC_W1_184)	32	RW	0000_0000h
15D0h	Peripheral Domain Access Control Word 0 (PDAC_W0_186)	32	RW	0000_0000h
15D4h	Peripheral Domain Access Control Word 1 (PDAC_W1_186)	32	RW	0000_0000h
15D8h	Peripheral Domain Access Control Word 0 (PDAC_W0_187)	32	RW	0000_0000h
15DCh	Peripheral Domain Access Control Word 1 (PDAC_W1_187)	32	RW	0000_0000h
15E0h	Peripheral Domain Access Control Word 0 (PDAC_W0_188)	32	RW	0000_0000h
15E4h	Peripheral Domain Access Control Word 1 (PDAC_W1_188)	32	RW	0000_0000h
15F8h	Peripheral Domain Access Control Word 0 (PDAC_W0_191)	32	RW	0000_0000h
15FCh	Peripheral Domain Access Control Word 1 (PDAC_W1_191)	32	RW	0000_0000h
1608h	Peripheral Domain Access Control Word 0 (PDAC_W0_193)	32	RW	0000_0000h
160Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_193)	32	RW	0000_0000h
1610h	Peripheral Domain Access Control Word 0 (PDAC_W0_194)	32	RW	0000_0000h
1614h	Peripheral Domain Access Control Word 1 (PDAC_W1_194)	32	RW	0000_0000h
1618h	Peripheral Domain Access Control Word 0 (PDAC_W0_195)	32	RW	0000_0000h
161Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_195)	32	RW	0000_0000h
1620h	Peripheral Domain Access Control Word 0 (PDAC_W0_196)	32	RW	0000_0000h
1624h	Peripheral Domain Access Control Word 1 (PDAC_W1_196)	32	RW	0000_0000h
1628h	Peripheral Domain Access Control Word 0 (PDAC_W0_197)	32	RW	0000_0000h
162Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_197)	32	RW	0000_0000h
1630h	Peripheral Domain Access Control Word 0 (PDAC_W0_198)	32	RW	0000_0000h
1634h	Peripheral Domain Access Control Word 1 (PDAC_W1_198)	32	RW	0000_0000h
1648h	Peripheral Domain Access Control Word 0 (PDAC_W0_201)	32	RW	0000_0000h
164Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_201)	32	RW	0000_0000h
1650h	Peripheral Domain Access Control Word 0 (PDAC_W0_202)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1654h	Peripheral Domain Access Control Word 1 (PDAC_W1_202)	32	RW	0000_0000h
1658h	Peripheral Domain Access Control Word 0 (PDAC_W0_203)	32	RW	0000_0000h
165Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_203)	32	RW	0000_0000h
1660h	Peripheral Domain Access Control Word 0 (PDAC_W0_204)	32	RW	0000_0000h
1664h	Peripheral Domain Access Control Word 1 (PDAC_W1_204)	32	RW	0000_0000h
1668h	Peripheral Domain Access Control Word 0 (PDAC_W0_205)	32	RW	0000_0000h
166Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_205)	32	RW	0000_0000h
1670h	Peripheral Domain Access Control Word 0 (PDAC_W0_206)	32	RW	0000_0000h
1674h	Peripheral Domain Access Control Word 1 (PDAC_W1_206)	32	RW	0000_0000h
1678h	Peripheral Domain Access Control Word 0 (PDAC_W0_207)	32	RW	0000_0000h
167Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_207)	32	RW	0000_0000h
1680h	Peripheral Domain Access Control Word 0 (PDAC_W0_208)	32	RW	0000_0000h
1684h	Peripheral Domain Access Control Word 1 (PDAC_W1_208)	32	RW	0000_0000h
1688h	Peripheral Domain Access Control Word 0 (PDAC_W0_209)	32	RW	0000_0000h
168Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_209)	32	RW	0000_0000h
16A0h	Peripheral Domain Access Control Word 0 (PDAC_W0_212)	32	RW	0000_0000h
16A4h	Peripheral Domain Access Control Word 1 (PDAC_W1_212)	32	RW	0000_0000h
16A8h	Peripheral Domain Access Control Word 0 (PDAC_W0_213)	32	RW	0000_0000h
16ACh	Peripheral Domain Access Control Word 1 (PDAC_W1_213)	32	RW	0000_0000h
16B0h	Peripheral Domain Access Control Word 0 (PDAC_W0_214)	32	RW	0000_0000h
16B4h	Peripheral Domain Access Control Word 1 (PDAC_W1_214)	32	RW	0000_0000h
16B8h	Peripheral Domain Access Control Word 0 (PDAC_W0_215)	32	RW	0000_0000h
16BCh	Peripheral Domain Access Control Word 1 (PDAC_W1_215)	32	RW	0000_0000h
16C0h	Peripheral Domain Access Control Word 0 (PDAC_W0_216)	32	RW	0000_0000h
16C4h	Peripheral Domain Access Control Word 1 (PDAC_W1_216)	32	RW	0000_0000h
16C8h	Peripheral Domain Access Control Word 0 (PDAC_W0_217)	32	RW	0000_0000h
16CCh	Peripheral Domain Access Control Word 1 (PDAC_W1_217)	32	RW	0000_0000h
16D8h	Peripheral Domain Access Control Word 0 (PDAC_W0_219)	32	RW	0000_0000h
16DCh	Peripheral Domain Access Control Word 1 (PDAC_W1_219)	32	RW	0000_0000h
16E0h	Peripheral Domain Access Control Word 0 (PDAC_W0_220)	32	RW	0000_0000h
16E4h	Peripheral Domain Access Control Word 1 (PDAC_W1_220)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
16E8h	Peripheral Domain Access Control Word 0 (PDAC_W0_221)	32	RW	0000_0000h
16ECh	Peripheral Domain Access Control Word 1 (PDAC_W1_221)	32	RW	0000_0000h
16F8h	Peripheral Domain Access Control Word 0 (PDAC_W0_223)	32	RW	0000_0000h
16FCh	Peripheral Domain Access Control Word 1 (PDAC_W1_223)	32	RW	0000_0000h
1700h	Peripheral Domain Access Control Word 0 (PDAC_W0_224)	32	RW	0000_0000h
1704h	Peripheral Domain Access Control Word 1 (PDAC_W1_224)	32	RW	0000_0000h
1708h	Peripheral Domain Access Control Word 0 (PDAC_W0_225)	32	RW	0000_0000h
170Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_225)	32	RW	0000_0000h
1710h	Peripheral Domain Access Control Word 0 (PDAC_W0_226)	32	RW	0000_0000h
1714h	Peripheral Domain Access Control Word 1 (PDAC_W1_226)	32	RW	0000_0000h
1718h	Peripheral Domain Access Control Word 0 (PDAC_W0_227)	32	RW	0000_0000h
171Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_227)	32	RW	0000_0000h
1728h	Peripheral Domain Access Control Word 0 (PDAC_W0_229)	32	RW	0000_0000h
172Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_229)	32	RW	0000_0000h
1738h	Peripheral Domain Access Control Word 0 (PDAC_W0_231)	32	RW	0000_0000h
173Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_231)	32	RW	0000_0000h
1740h	Peripheral Domain Access Control Word 0 (PDAC_W0_232)	32	RW	0000_0000h
1744h	Peripheral Domain Access Control Word 1 (PDAC_W1_232)	32	RW	0000_0000h
1760h	Peripheral Domain Access Control Word 0 (PDAC_W0_236)	32	RW	0000_0000h
1764h	Peripheral Domain Access Control Word 1 (PDAC_W1_236)	32	RW	0000_0000h
1800h	Peripheral Domain Access Control Word 0 (PDAC_W0_256)	32	RW	0000_0000h
1804h	Peripheral Domain Access Control Word 1 (PDAC_W1_256)	32	RW	0000_0000h
1808h	Peripheral Domain Access Control Word 0 (PDAC_W0_257)	32	RW	0000_0000h
180Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_257)	32	RW	0000_0000h
1820h	Peripheral Domain Access Control Word 0 (PDAC_W0_260)	32	RW	0000_0000h
1824h	Peripheral Domain Access Control Word 1 (PDAC_W1_260)	32	RW	0000_0000h
1828h	Peripheral Domain Access Control Word 0 (PDAC_W0_261)	32	RW	0000_0000h
182Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_261)	32	RW	0000_0000h
1830h	Peripheral Domain Access Control Word 0 (PDAC_W0_262)	32	RW	0000_0000h
1834h	Peripheral Domain Access Control Word 1 (PDAC_W1_262)	32	RW	0000_0000h
1838h	Peripheral Domain Access Control Word 0 (PDAC_W0_263)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
183Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_263)	32	RW	0000_0000h
1840h	Peripheral Domain Access Control Word 0 (PDAC_W0_264)	32	RW	0000_0000h
1844h	Peripheral Domain Access Control Word 1 (PDAC_W1_264)	32	RW	0000_0000h
1848h	Peripheral Domain Access Control Word 0 (PDAC_W0_265)	32	RW	0000_0000h
184Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_265)	32	RW	0000_0000h
1850h	Peripheral Domain Access Control Word 0 (PDAC_W0_266)	32	RW	0000_0000h
1854h	Peripheral Domain Access Control Word 1 (PDAC_W1_266)	32	RW	0000_0000h
1858h	Peripheral Domain Access Control Word 0 (PDAC_W0_267)	32	RW	0000_0000h
185Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_267)	32	RW	0000_0000h
1860h	Peripheral Domain Access Control Word 0 (PDAC_W0_268)	32	RW	0000_0000h
1864h	Peripheral Domain Access Control Word 1 (PDAC_W1_268)	32	RW	0000_0000h
1868h	Peripheral Domain Access Control Word 0 (PDAC_W0_269)	32	RW	0000_0000h
186Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_269)	32	RW	0000_0000h
1870h	Peripheral Domain Access Control Word 0 (PDAC_W0_270)	32	RW	0000_0000h
1874h	Peripheral Domain Access Control Word 1 (PDAC_W1_270)	32	RW	0000_0000h
1878h	Peripheral Domain Access Control Word 0 (PDAC_W0_271)	32	RW	0000_0000h
187Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_271)	32	RW	0000_0000h
1880h	Peripheral Domain Access Control Word 0 (PDAC_W0_272)	32	RW	0000_0000h
1884h	Peripheral Domain Access Control Word 1 (PDAC_W1_272)	32	RW	0000_0000h
1888h	Peripheral Domain Access Control Word 0 (PDAC_W0_273)	32	RW	0000_0000h
188Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_273)	32	RW	0000_0000h
1890h	Peripheral Domain Access Control Word 0 (PDAC_W0_274)	32	RW	0000_0000h
1894h	Peripheral Domain Access Control Word 1 (PDAC_W1_274)	32	RW	0000_0000h
1898h	Peripheral Domain Access Control Word 0 (PDAC_W0_275)	32	RW	0000_0000h
189Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_275)	32	RW	0000_0000h
18A0h	Peripheral Domain Access Control Word 0 (PDAC_W0_276)	32	RW	0000_0000h
18A4h	Peripheral Domain Access Control Word 1 (PDAC_W1_276)	32	RW	0000_0000h
18A8h	Peripheral Domain Access Control Word 0 (PDAC_W0_277)	32	RW	0000_0000h
18ACh	Peripheral Domain Access Control Word 1 (PDAC_W1_277)	32	RW	0000_0000h
18B0h	Peripheral Domain Access Control Word 0 (PDAC_W0_278)	32	RW	0000_0000h
18B4h	Peripheral Domain Access Control Word 1 (PDAC_W1_278)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
18B8h	Peripheral Domain Access Control Word 0 (PDAC_W0_279)	32	RW	0000_0000h
18BCh	Peripheral Domain Access Control Word 1 (PDAC_W1_279)	32	RW	0000_0000h
18C0h	Peripheral Domain Access Control Word 0 (PDAC_W0_280)	32	RW	0000_0000h
18C4h	Peripheral Domain Access Control Word 1 (PDAC_W1_280)	32	RW	0000_0000h
18C8h	Peripheral Domain Access Control Word 0 (PDAC_W0_281)	32	RW	0000_0000h
18CCh	Peripheral Domain Access Control Word 1 (PDAC_W1_281)	32	RW	0000_0000h
18D8h	Peripheral Domain Access Control Word 0 (PDAC_W0_283)	32	RW	0000_0000h
18DCh	Peripheral Domain Access Control Word 1 (PDAC_W1_283)	32	RW	0000_0000h
18E8h	Peripheral Domain Access Control Word 0 (PDAC_W0_285)	32	RW	0000_0000h
18ECh	Peripheral Domain Access Control Word 1 (PDAC_W1_285)	32	RW	0000_0000h
1900h	Peripheral Domain Access Control Word 0 (PDAC_W0_288)	32	RW	0000_0000h
1904h	Peripheral Domain Access Control Word 1 (PDAC_W1_288)	32	RW	0000_0000h
1918h	Peripheral Domain Access Control Word 0 (PDAC_W0_291)	32	RW	0000_0000h
191Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_291)	32	RW	0000_0000h
1920h	Peripheral Domain Access Control Word 0 (PDAC_W0_292)	32	RW	0000_0000h
1924h	Peripheral Domain Access Control Word 1 (PDAC_W1_292)	32	RW	0000_0000h
1928h	Peripheral Domain Access Control Word 0 (PDAC_W0_293)	32	RW	0000_0000h
192Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_293)	32	RW	0000_0000h
1930h	Peripheral Domain Access Control Word 0 (PDAC_W0_294)	32	RW	0000_0000h
1934h	Peripheral Domain Access Control Word 1 (PDAC_W1_294)	32	RW	0000_0000h
1938h	Peripheral Domain Access Control Word 0 (PDAC_W0_295)	32	RW	0000_0000h
193Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_295)	32	RW	0000_0000h
1940h	Peripheral Domain Access Control Word 0 (PDAC_W0_296)	32	RW	0000_0000h
1944h	Peripheral Domain Access Control Word 1 (PDAC_W1_296)	32	RW	0000_0000h
1948h	Peripheral Domain Access Control Word 0 (PDAC_W0_297)	32	RW	0000_0000h
194Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_297)	32	RW	0000_0000h
1950h	Peripheral Domain Access Control Word 0 (PDAC_W0_298)	32	RW	0000_0000h
1954h	Peripheral Domain Access Control Word 1 (PDAC_W1_298)	32	RW	0000_0000h
1978h	Peripheral Domain Access Control Word 0 (PDAC_W0_303)	32	RW	0000_0000h
197Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_303)	32	RW	0000_0000h
1980h	Peripheral Domain Access Control Word 0 (PDAC_W0_304)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1984h	Peripheral Domain Access Control Word 1 (PDAC_W1_304)	32	RW	0000_0000h
1998h	Peripheral Domain Access Control Word 0 (PDAC_W0_307)	32	RW	0000_0000h
199Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_307)	32	RW	0000_0000h
19B8h	Peripheral Domain Access Control Word 0 (PDAC_W0_311)	32	RW	0000_0000h
19BCh	Peripheral Domain Access Control Word 1 (PDAC_W1_311)	32	RW	0000_0000h
19D0h	Peripheral Domain Access Control Word 0 (PDAC_W0_314)	32	RW	0000_0000h
19D4h	Peripheral Domain Access Control Word 1 (PDAC_W1_314)	32	RW	0000_0000h
19D8h	Peripheral Domain Access Control Word 0 (PDAC_W0_315)	32	RW	0000_0000h
19DCh	Peripheral Domain Access Control Word 1 (PDAC_W1_315)	32	RW	0000_0000h
2000h	Memory Region Descriptor Word 0 (MRGD_W0_0)	32	RW	0000_0001h
2004h	Memory Region Descriptor Word 1 (MRGD_W1_0)	32	RW	0000_001Fh
2008h	Memory Region Descriptor Word 2 (MRGD_W2_0)	32	RW	0000_0000h
200Ch	Memory Region Descriptor Word 3 (MRGD_W3_0)	32	RW	0000_0000h
2020h	Memory Region Descriptor Word 0 (MRGD_W0_1)	32	RW	0000_0001h
2024h	Memory Region Descriptor Word 1 (MRGD_W1_1)	32	RW	0000_001Fh
2028h	Memory Region Descriptor Word 2 (MRGD_W2_1)	32	RW	0000_0000h
202Ch	Memory Region Descriptor Word 3 (MRGD_W3_1)	32	RW	0000_0000h
2040h	Memory Region Descriptor Word 0 (MRGD_W0_2)	32	RW	0000_0001h
2044h	Memory Region Descriptor Word 1 (MRGD_W1_2)	32	RW	0000_001Fh
2048h	Memory Region Descriptor Word 2 (MRGD_W2_2)	32	RW	0000_0000h
204Ch	Memory Region Descriptor Word 3 (MRGD_W3_2)	32	RW	0000_0000h
2060h	Memory Region Descriptor Word 0 (MRGD_W0_3)	32	RW	0000_0001h
2064h	Memory Region Descriptor Word 1 (MRGD_W1_3)	32	RW	0000_001Fh
2068h	Memory Region Descriptor Word 2 (MRGD_W2_3)	32	RW	0000_0000h
206Ch	Memory Region Descriptor Word 3 (MRGD_W3_3)	32	RW	0000_0000h
2080h	Memory Region Descriptor Word 0 (MRGD_W0_4)	32	RW	0000_0001h
2084h	Memory Region Descriptor Word 1 (MRGD_W1_4)	32	RW	0000_001Fh
2088h	Memory Region Descriptor Word 2 (MRGD_W2_4)	32	RW	0000_0000h
208Ch	Memory Region Descriptor Word 3 (MRGD_W3_4)	32	RW	0000_0000h
20A0h	Memory Region Descriptor Word 0 (MRGD_W0_5)	32	RW	0000_0001h
20A4h	Memory Region Descriptor Word 1 (MRGD_W1_5)	32	RW	0000_001Fh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20A8h	Memory Region Descriptor Word 2 (MRGD_W2_5)	32	RW	0000_0000h
20ACh	Memory Region Descriptor Word 3 (MRGD_W3_5)	32	RW	0000_0000h
20C0h	Memory Region Descriptor Word 0 (MRGD_W0_6)	32	RW	0000_0001h
20C4h	Memory Region Descriptor Word 1 (MRGD_W1_6)	32	RW	0000_001Fh
20C8h	Memory Region Descriptor Word 2 (MRGD_W2_6)	32	RW	0000_0000h
20CCh	Memory Region Descriptor Word 3 (MRGD_W3_6)	32	RW	0000_0000h
20E0h	Memory Region Descriptor Word 0 (MRGD_W0_7)	32	RW	0000_0001h
20E4h	Memory Region Descriptor Word 1 (MRGD_W1_7)	32	RW	0000_001Fh
20E8h	Memory Region Descriptor Word 2 (MRGD_W2_7)	32	RW	0000_0000h
20ECh	Memory Region Descriptor Word 3 (MRGD_W3_7)	32	RW	0000_0000h
2100h	Memory Region Descriptor Word 0 (MRGD_W0_8)	32	RW	0000_0001h
2104h	Memory Region Descriptor Word 1 (MRGD_W1_8)	32	RW	0000_001Fh
2108h	Memory Region Descriptor Word 2 (MRGD_W2_8)	32	RW	0000_0000h
210Ch	Memory Region Descriptor Word 3 (MRGD_W3_8)	32	RW	0000_0000h
2120h	Memory Region Descriptor Word 0 (MRGD_W0_9)	32	RW	0000_0001h
2124h	Memory Region Descriptor Word 1 (MRGD_W1_9)	32	RW	0000_001Fh
2128h	Memory Region Descriptor Word 2 (MRGD_W2_9)	32	RW	0000_0000h
212Ch	Memory Region Descriptor Word 3 (MRGD_W3_9)	32	RW	0000_0000h
2140h	Memory Region Descriptor Word 0 (MRGD_W0_10)	32	RW	0000_0001h
2144h	Memory Region Descriptor Word 1 (MRGD_W1_10)	32	RW	0000_001Fh
2148h	Memory Region Descriptor Word 2 (MRGD_W2_10)	32	RW	0000_0000h
214Ch	Memory Region Descriptor Word 3 (MRGD_W3_10)	32	RW	0000_0000h
2160h	Memory Region Descriptor Word 0 (MRGD_W0_11)	32	RW	0000_0001h
2164h	Memory Region Descriptor Word 1 (MRGD_W1_11)	32	RW	0000_001Fh
2168h	Memory Region Descriptor Word 2 (MRGD_W2_11)	32	RW	0000_0000h
216Ch	Memory Region Descriptor Word 3 (MRGD_W3_11)	32	RW	0000_0000h
2180h	Memory Region Descriptor Word 0 (MRGD_W0_12)	32	RW	0000_0001h
2184h	Memory Region Descriptor Word 1 (MRGD_W1_12)	32	RW	0000_001Fh
2188h	Memory Region Descriptor Word 2 (MRGD_W2_12)	32	RW	0000_0000h
218Ch	Memory Region Descriptor Word 3 (MRGD_W3_12)	32	RW	0000_0000h
21A0h	Memory Region Descriptor Word 0 (MRGD_W0_13)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
21A4h	Memory Region Descriptor Word 1 (MRGD_W1_13)	32	RW	0000_001Fh
21A8h	Memory Region Descriptor Word 2 (MRGD_W2_13)	32	RW	0000_0000h
21ACh	Memory Region Descriptor Word 3 (MRGD_W3_13)	32	RW	0000_0000h
21C0h	Memory Region Descriptor Word 0 (MRGD_W0_14)	32	RW	0000_0001h
21C4h	Memory Region Descriptor Word 1 (MRGD_W1_14)	32	RW	0000_001Fh
21C8h	Memory Region Descriptor Word 2 (MRGD_W2_14)	32	RW	0000_0000h
21CCh	Memory Region Descriptor Word 3 (MRGD_W3_14)	32	RW	0000_0000h
21E0h	Memory Region Descriptor Word 0 (MRGD_W0_15)	32	RW	0000_0001h
21E4h	Memory Region Descriptor Word 1 (MRGD_W1_15)	32	RW	0000_001Fh
21E8h	Memory Region Descriptor Word 2 (MRGD_W2_15)	32	RW	0000_0000h
21ECh	Memory Region Descriptor Word 3 (MRGD_W3_15)	32	RW	0000_0000h
2200h	Memory Region Descriptor Word 0 (MRGD_W0_16)	32	RW	0000_0001h
2204h	Memory Region Descriptor Word 1 (MRGD_W1_16)	32	RW	0000_001Fh
2208h	Memory Region Descriptor Word 2 (MRGD_W2_16)	32	RW	0000_0000h
220Ch	Memory Region Descriptor Word 3 (MRGD_W3_16)	32	RW	0000_0000h
2220h	Memory Region Descriptor Word 0 (MRGD_W0_17)	32	RW	0000_0001h
2224h	Memory Region Descriptor Word 1 (MRGD_W1_17)	32	RW	0000_001Fh
2228h	Memory Region Descriptor Word 2 (MRGD_W2_17)	32	RW	0000_0000h
222Ch	Memory Region Descriptor Word 3 (MRGD_W3_17)	32	RW	0000_0000h
2240h	Memory Region Descriptor Word 0 (MRGD_W0_18)	32	RW	0000_0001h
2244h	Memory Region Descriptor Word 1 (MRGD_W1_18)	32	RW	0000_001Fh
2248h	Memory Region Descriptor Word 2 (MRGD_W2_18)	32	RW	0000_0000h
224Ch	Memory Region Descriptor Word 3 (MRGD_W3_18)	32	RW	0000_0000h
2260h	Memory Region Descriptor Word 0 (MRGD_W0_19)	32	RW	0000_0001h
2264h	Memory Region Descriptor Word 1 (MRGD_W1_19)	32	RW	0000_001Fh
2268h	Memory Region Descriptor Word 2 (MRGD_W2_19)	32	RW	0000_0000h
226Ch	Memory Region Descriptor Word 3 (MRGD_W3_19)	32	RW	0000_0000h
2280h	Memory Region Descriptor Word 0 (MRGD_W0_20)	32	RW	0000_0001h
2284h	Memory Region Descriptor Word 1 (MRGD_W1_20)	32	RW	0000_001Fh
2288h	Memory Region Descriptor Word 2 (MRGD_W2_20)	32	RW	0000_0000h
228Ch	Memory Region Descriptor Word 3 (MRGD_W3_20)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
22A0h	Memory Region Descriptor Word 0 (MRGD_W0_21)	32	RW	0000_0001h
22A4h	Memory Region Descriptor Word 1 (MRGD_W1_21)	32	RW	0000_001Fh
22A8h	Memory Region Descriptor Word 2 (MRGD_W2_21)	32	RW	0000_0000h
22ACh	Memory Region Descriptor Word 3 (MRGD_W3_21)	32	RW	0000_0000h
22C0h	Memory Region Descriptor Word 0 (MRGD_W0_22)	32	RW	0000_0001h
22C4h	Memory Region Descriptor Word 1 (MRGD_W1_22)	32	RW	0000_001Fh
22C8h	Memory Region Descriptor Word 2 (MRGD_W2_22)	32	RW	0000_0000h
22CCh	Memory Region Descriptor Word 3 (MRGD_W3_22)	32	RW	0000_0000h
22E0h	Memory Region Descriptor Word 0 (MRGD_W0_23)	32	RW	0000_0001h
22E4h	Memory Region Descriptor Word 1 (MRGD_W1_23)	32	RW	0000_001Fh
22E8h	Memory Region Descriptor Word 2 (MRGD_W2_23)	32	RW	0000_0000h
22ECh	Memory Region Descriptor Word 3 (MRGD_W3_23)	32	RW	0000_0000h
2300h	Memory Region Descriptor Word 0 (MRGD_W0_24)	32	RW	0000_0001h
2304h	Memory Region Descriptor Word 1 (MRGD_W1_24)	32	RW	0000_001Fh
2308h	Memory Region Descriptor Word 2 (MRGD_W2_24)	32	RW	0000_0000h
230Ch	Memory Region Descriptor Word 3 (MRGD_W3_24)	32	RW	0000_0000h
2320h	Memory Region Descriptor Word 0 (MRGD_W0_25)	32	RW	0000_0001h
2324h	Memory Region Descriptor Word 1 (MRGD_W1_25)	32	RW	0000_001Fh
2328h	Memory Region Descriptor Word 2 (MRGD_W2_25)	32	RW	0000_0000h
232Ch	Memory Region Descriptor Word 3 (MRGD_W3_25)	32	RW	0000_0000h
2340h	Memory Region Descriptor Word 0 (MRGD_W0_26)	32	RW	0000_0001h
2344h	Memory Region Descriptor Word 1 (MRGD_W1_26)	32	RW	0000_001Fh
2348h	Memory Region Descriptor Word 2 (MRGD_W2_26)	32	RW	0000_0000h
234Ch	Memory Region Descriptor Word 3 (MRGD_W3_26)	32	RW	0000_0000h
2360h	Memory Region Descriptor Word 0 (MRGD_W0_27)	32	RW	0000_0001h
2364h	Memory Region Descriptor Word 1 (MRGD_W1_27)	32	RW	0000_001Fh
2368h	Memory Region Descriptor Word 2 (MRGD_W2_27)	32	RW	0000_0000h
236Ch	Memory Region Descriptor Word 3 (MRGD_W3_27)	32	RW	0000_0000h
2380h	Memory Region Descriptor Word 0 (MRGD_W0_28)	32	RW	0000_0001h
2384h	Memory Region Descriptor Word 1 (MRGD_W1_28)	32	RW	0000_001Fh
2388h	Memory Region Descriptor Word 2 (MRGD_W2_28)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
238Ch	Memory Region Descriptor Word 3 (MRGD_W3_28)	32	RW	0000_0000h
23A0h	Memory Region Descriptor Word 0 (MRGD_W0_29)	32	RW	0000_0001h
23A4h	Memory Region Descriptor Word 1 (MRGD_W1_29)	32	RW	0000_001Fh
23A8h	Memory Region Descriptor Word 2 (MRGD_W2_29)	32	RW	0000_0000h
23ACh	Memory Region Descriptor Word 3 (MRGD_W3_29)	32	RW	0000_0000h
23C0h	Memory Region Descriptor Word 0 (MRGD_W0_30)	32	RW	0000_0001h
23C4h	Memory Region Descriptor Word 1 (MRGD_W1_30)	32	RW	0000_001Fh
23C8h	Memory Region Descriptor Word 2 (MRGD_W2_30)	32	RW	0000_0000h
23CCh	Memory Region Descriptor Word 3 (MRGD_W3_30)	32	RW	0000_0000h
23E0h	Memory Region Descriptor Word 0 (MRGD_W0_31)	32	RW	0000_0001h
23E4h	Memory Region Descriptor Word 1 (MRGD_W1_31)	32	RW	0000_001Fh
23E8h	Memory Region Descriptor Word 2 (MRGD_W2_31)	32	RW	0000_0000h
23ECh	Memory Region Descriptor Word 3 (MRGD_W3_31)	32	RW	0000_0000h
2400h	Memory Region Descriptor Word 0 (MRGD_W0_32)	32	RW	0000_0001h
2404h	Memory Region Descriptor Word 1 (MRGD_W1_32)	32	RW	0000_001Fh
2408h	Memory Region Descriptor Word 2 (MRGD_W2_32)	32	RW	0000_0000h
240Ch	Memory Region Descriptor Word 3 (MRGD_W3_32)	32	RW	0000_0000h
2420h	Memory Region Descriptor Word 0 (MRGD_W0_33)	32	RW	0000_0001h
2424h	Memory Region Descriptor Word 1 (MRGD_W1_33)	32	RW	0000_001Fh
2428h	Memory Region Descriptor Word 2 (MRGD_W2_33)	32	RW	0000_0000h
242Ch	Memory Region Descriptor Word 3 (MRGD_W3_33)	32	RW	0000_0000h
2440h	Memory Region Descriptor Word 0 (MRGD_W0_34)	32	RW	0000_0001h
2444h	Memory Region Descriptor Word 1 (MRGD_W1_34)	32	RW	0000_001Fh
2448h	Memory Region Descriptor Word 2 (MRGD_W2_34)	32	RW	0000_0000h
244Ch	Memory Region Descriptor Word 3 (MRGD_W3_34)	32	RW	0000_0000h
2460h	Memory Region Descriptor Word 0 (MRGD_W0_35)	32	RW	0000_0001h
2464h	Memory Region Descriptor Word 1 (MRGD_W1_35)	32	RW	0000_001Fh
2468h	Memory Region Descriptor Word 2 (MRGD_W2_35)	32	RW	0000_0000h
246Ch	Memory Region Descriptor Word 3 (MRGD_W3_35)	32	RW	0000_0000h

18.8.3.2 Control (CR)

Offset

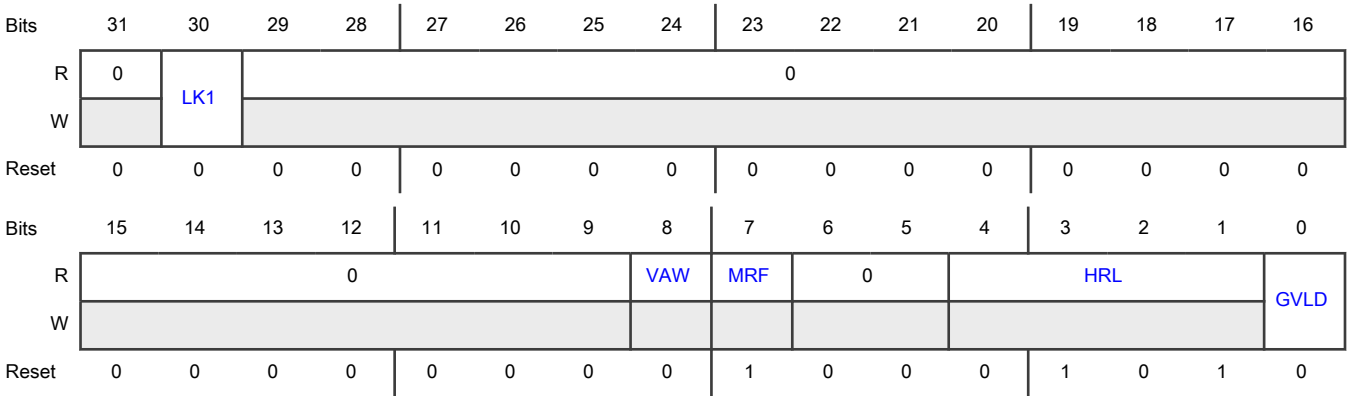
Register	Offset
CR	0h

Function

Provides XRDC status and enables XRDC operation.

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 —	Reserved
30 LK1	Lock Prohibits writes to this register. • If unlocked, this register accepts any secure privileged write. • If locked, you cannot write to this register and it remains read-only until after the next reset. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Unlocked 1b - Locks When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Locks
29-9 —	Reserved
8 VAW	Virtualization Aware Indicates whether domain assignments support the optional inclusion of a logical partition identifier, which is also known as an operating system number or Arm virtual machine identifier (VMID). 0b - Not virtualization-aware 1b - Virtualization-aware
7 MRF	Memory Region Format Indicates the format of memory region descriptors. 0b - Reserved 1b - SMPU family format
6-5 —	Reserved
4-1 HRL	Hardware Revision Level Indicates the XRDC hardware revision level, which is associated with a set of functional characteristics of the module.
0 GVLD	Global Valid (XRDC Global Enable/Disable) Enables XRDC. When XRDC is disabled, all bus masters can access all targets. 0b - Disables 1b - Enables

18.8.3.3 Hardware Configuration 0 (HWCFG0)

Offset

Register	Offset
HWCFG0	F0h

Function

Indicates XRDC configuration details, including:

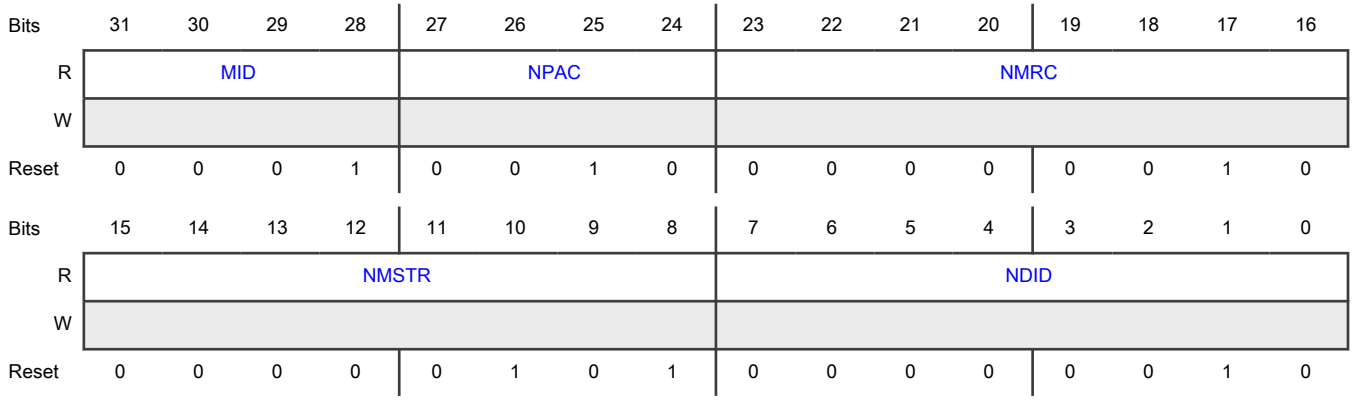
- XRDC module ID
- Number of implemented domains

- Number of bus masters
- Number of MRCs
- Number of PACs

Attempting to write to this register causes an error.

Access: Secure privileged read

Diagram



Fields

Field	Function
31-28 MID	Module ID
27-24 NPAC	Number Of PACs Indicates the number of PACs minus 1. In other words, the actual number of PACs is NPAC + 1.
23-16 NMRC	Number of MRCs Indicates the number of MRCs minus 1. In other words, the actual number of MRCs is NMRC + 1.
15-8 NMSTR	Number Of Bus Masters Indicates the number of bus masters minus 1. In other words, the actual number of bus masters is NMSTR + 1.
7-0 NDID	Number Of DIDs Indicates the number of domains (DIDs) minus 1. In other words, the actual number of DIDs is NDID + 1.

18.8.3.4 Hardware Configuration 1 (HWCFG1)

Offset

Register	Offset
HWCFG1	F4h

Function

Indicates the DID of the bus master making the current transaction request. See Domain error capture management for information about typical usage.

Attempting to write to this register causes an error.

Access: Secure privileged read

Diagram

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

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

1. The reset value is determined by the current configuration of the accessing master.

Fields

Field	Function
31-4 —	Reserved
3-0 DID	Domain Identifier Indicates the DID of the requesting bus master.

18.8.3.5 Hardware Configuration 2 (HWCFG2)

Offset

Register	Offset
HWCFG2	F8h

Function

For masters 0–31, indicates whether a given master has a built-in PID register as part of its programming model. If not, you must use the corresponding PIDm register to mimic the functionality of a built-in PID register.

Each bit corresponds to the same numbered master. For example, if PIDP18 is 1, bus master 18 has its own PID register. If PIDP18 is 0, then master 18 does not have its own PID register and you must use PID18.

Attempting to write to this register causes an error.

Access: Secure privileged read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PIDP31	PIDP30	PIDP29	PIDP28	PIDP27	PIDP26	PIDP25	PIDP24	PIDP23	PIDP22	PIDP21	PIDP20	PIDP19	PIDP18	PIDP17	PIDP16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PIDP15	PIDP14	PIDP13	PIDP12	PIDP11	PIDP10	PIDP9	PIDP8	PIDP7	PIDP6	PIDP5	PIDP4	PIDP3	PIDP2	PIDP1	PIDP0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PIDPn	Process Identifier Present 0b - Does not have PID register 1b - Has PID register

18.8.3.6 Master Domain Assignment Configuration (MDACFG0 - MDACFG5)

Offset

Register	Offset
MDACFG0	100h
MDACFG1	101h
MDACFG2	102h
MDACFG3	103h
MDACFG4	104h
MDACFG5	105h

Function

Indicates the number of implemented master domain assignment registers (MDA_W*w*_*m*_DFMT0 or MDA_W*w*_*m*_DFMT1) for master *m*, where *m* ranges from 0 to 63. You can read these registers using 8-, 16-, or 32-bit accesses.

If NMDAR is 0, the associated master does not exist.

Attempting to write to this register causes an error.

Access: Secure privileged read

Diagram

Bits	7	6	5	4	3	2	1	0
R	NCM	0			NMDAR			
W								
Reset	See Register reset values.							

Register reset values

Register	Reset value
MDACFG0	01h
MDACFG1–MDACFG2	81h
MDACFG3–MDACFG4	01h
MDACFG5	81h

Fields

Field	Function
7 NCM	Noncore Master If NMDAR is greater than zero, indicates whether master *m* uses MDA_W*w*_*m*_DFMT0 or MDA_W*w*_*m*_DFMT1 to configure domain assignment. This field is 0 for a non-existent master. 0b - Core master or master does not exist 1b - Noncore master
6-4 —	Reserved
3-0 NMDAR	Number Of Master Domain Assignment Registers Indicates the number of master domain assignment registers (MDA_W*w*_*m*_DFMT0 or MDA_W*w*_*m*_DFMT1) associated with master *m*. 0000b - Master does not exist 0001b-1000b - Number of registers All other values are reserved.

18.8.3.7 Memory Region Configuration (MRCFG0 - MRCFG2)

Offset

Register	Offset
MRCFG0	140h
MRCFG1	141h
MRCFG2	142h

Function

Indicates the number of memory region descriptors (*r*) for MRC*c*, from 4 to 16 in increments of four, with 0 indicating a non-existent MRC. These registers are organized as byte-sized data arrays and can be read using 8-, 16-, or 32-bit accesses.

Attempting to write to this register causes an error.

Access: Secure Privileged Read

Diagram

Bits	7	6	5	4	3	2	1	0
R	0				NMRGD			
W								
Reset	See Register reset values.							

Register reset values

Register	Reset value
MRCFG0–MRCFG1	10h
MRCFG2	04h

Fields

Field	Function
7-5 —	Reserved
4-0 NMRGD	Number Of Memory Region Descriptors Indicates the number of memory region descriptors associated with the MRC. 0_0000b - MRC does not exist 0_0100b - 4 0_1000b - 8

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0_1100b - 12 1_0000b - 16 All other values are reserved.

18.8.3.8 Domain Error Location (DERRLOC0 - DERRLOC2)

Offset

Register	Offset
DERRLOC0	200h
DERRLOC1	204h
DERRLOC2	208h

Function

Indicates the MRC or PAC instance in domain *d* where an access violation has occurred. Each bit corresponds to the like-numbered instance. For more information, see Domain error capture management.

Attempting to write to this register causes an error.

Access: Secure privileged read

Diagram

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

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

Fields

Field	Function
31-20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-16 PACINST	PAC Instance Indicates the presence of a detected access violation for domain *d* in a PAC instance. Each bit corresponds to the instance index for the DERR_W*w*_*i* registers: bit 16 of the register corresponds to the DERR_W*w*_*16* registers, which show error information for PAC 0, and so on. Multiple bits can be 1 at any time, indicating access violations have been detected across multiple PACs. For each bit in this field: 0b - No access violation error or the PAC instance is not physically present 1b - Access violation detected
15-0 MRCINST	MRC Instance Indicates the presence of a detected access violation for domain *d* in an MRC instance. Each bit corresponds to the like-numbered MRC instance: bit 0 (bit 0 of the register) corresponds to MRC instance 0, and so on. Multiple bits can be 1 at any time, indicating access violations have been detected across multiple MRCs. For each bit in this field: 0b - No access violation error or the MRC instance is not physically present 1b - Access violation detected

18.8.3.9 Domain Error Word 0 (DERR_W0_0 - DERR_W0_18)

Offset

Register	Offset
DERR_W0_0	400h
DERR_W0_1	410h
DERR_W0_2	420h
DERR_W0_16	500h
DERR_W0_17	510h
DERR_W0_18	520h

Function

Indicates the address of an access violation detected by an MRC or a PAC, indexed by the MRC or PAC instance (*i*) that detected the violation, as indicated in DERRLOC*d*. This register is part of a 16-byte set:

- DERR_W0_*i*: Word 0, the first 4 bytes
- DERR_W1_*i*: Word 1, the second 4 bytes
- Word 2, 4 reserved bytes
- DERR_W3_*i*: Word 3, the fourth 4 bytes

The first 16 sets (*i* from 0 to 15) are associated with MRCs and the rest (starting with *i* = 16) are associated with PACs. For more information, see Domain error capture management.

The access violation exception handler for each domain has visibility only into the captured error information for that domain.

When XRDC detects an access violation, it captures the error information and disables subsequent updates to the error capture registers until you write to DERR_W3_*i*.

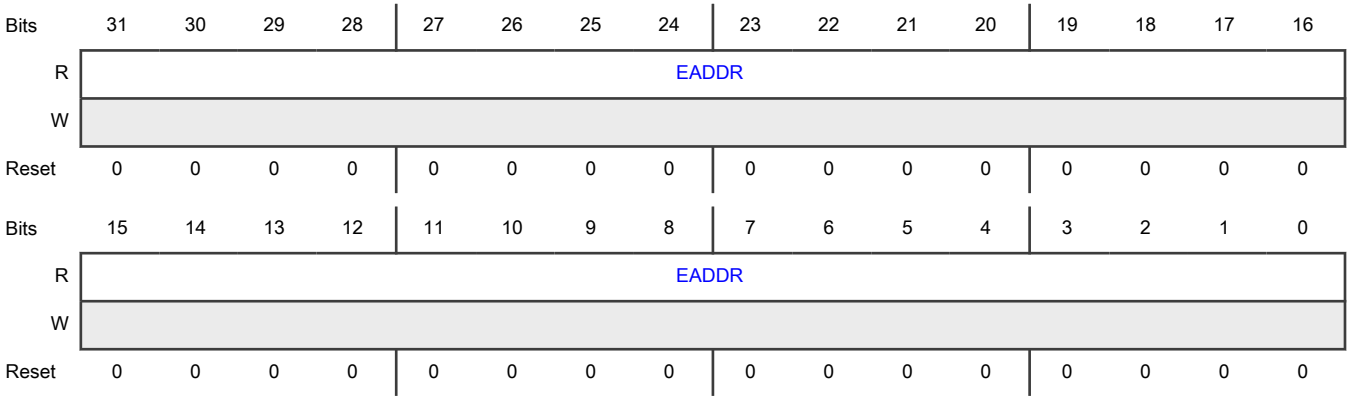
NOTE

If masters with the same DID cause simultaneous error accesses, the error capture registers record only the error of the lowest target index.

Attempting to write to this register causes an error. Attempting to read the error registers for a non-existent MRC or PAC instance causes an error.

Access: Secure privileged read

Diagram



Fields

Field	Function
31-0 EADDR	Error Address Indicates the target address of the first transaction that causes an access violation after reset or after rearming error capture.

18.8.3.10 Domain Error Word 1 (DERR_W1_0 - DERR_W1_18)

Offset

Register	Offset
DERR_W1_0	404h
DERR_W1_1	414h
DERR_W1_2	424h
DERR_W1_16	504h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
DERR_W1_17	514h
DERR_W1_18	524h

Function

Indicates the attributes of an access violation detected by an MRC or a PAC, indexed by the MRC or PAC instance (*i*) that detected the access violation, as indicated in DERRLOC*d*. For more information, see DERR_W0_*i* and Domain error capture management.

NOTE

If masters with the same DID cause simultaneous error accesses, the error capture registers record only the error of the lowest target index.

Attempting to write to this register causes an error. Attempting to read the error registers for a non-existent MRC or PAC instance causes an error.

Access: Secure privileged read

Diagram

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

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

Fields

Field	Function
31-30 EST	Error State Indicates the state of access violations for this domain in this instance of the MRC or PAC. After the first access violation to occur after reset or rearming of error capture, XRDC records subsequent errors as an overrun condition without any data captured. 00b-01b - No access violations detected 10b - A single access violation has been detected 11b - Multiple access violations have been detected
29-27 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
26-24 EPORT	Error Port Identifies the encoded port number of the MRC that detected the access violation. See the chip-specific configuration details for the MRC port connections. For access violations detected by a PAC, this field is zero.
23-12 —	Reserved
11 ERW	Error Read Or Write Indicates whether the captured access violation occurred on a read or write access. 0b - Read access 1b - Write access
10-8 EATR	Error Attributes Indicates attributes of the access violation. 000b - Secure user mode, instruction fetch access 001b - Secure user mode, data access 010b - Secure privileged mode, instruction fetch access 011b - Secure privileged mode, data access 100b - Nonsecure user mode, instruction fetch access 101b - Nonsecure user mode, data access 110b - Nonsecure privileged mode, instruction fetch access 111b - Nonsecure privileged mode, data access
7-4 —	Reserved
3-0 EDID	Error Domain Identifier Indicates the DID of the access violation.

18.8.3.11 Domain Error Word 3 (DERR_W3_0 - DERR_W3_18)

Offset

Register	Offset
DERR_W3_0	40Ch
DERR_W3_1	41Ch
DERR_W3_2	42Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
DERR_W3_16	50Ch
DERR_W3_17	51Ch
DERR_W3_18	52Ch

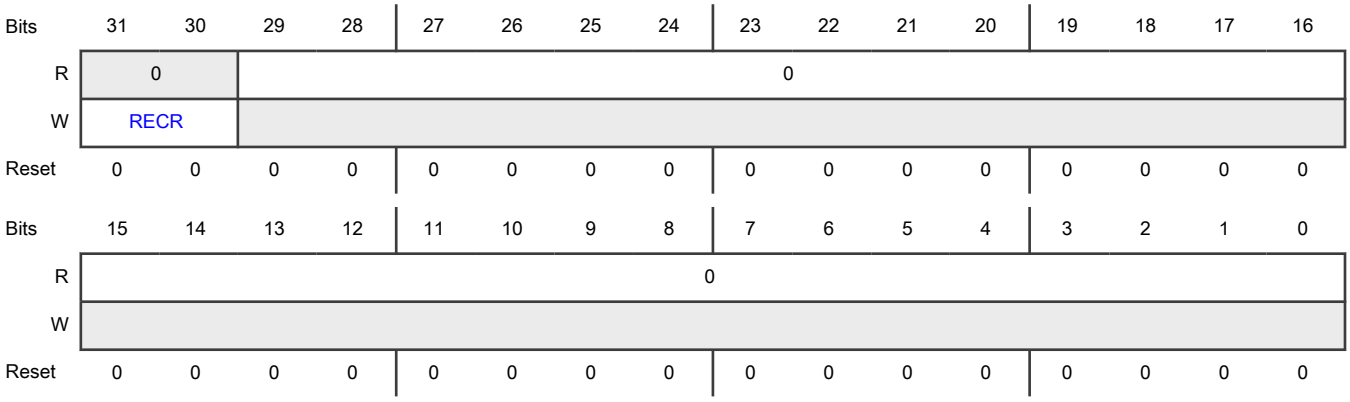
Function

Rearms instance error capture, resets the error capture registers (DERR_W0_*d*, DERR_W1_*d*), and deasserts the instance bit in DERRLOC*d*. After reading the access violation error information, an error handler must write 1 to RECR.

This register returns 0000h when read. Attempted reads of an MRC or PAC instance that is not physically present cause an error.

For more information, see Domain error capture management.

Diagram



Fields

Field	Function
31-30 RECR	Rearm Error Capture Registers Resets and rearms the domain error capture registers for this instance, including deasserting the instance bit in DERRLOC*d*. 00b,10b,11b - No effect 01b - Rearms error capture, resets error capture registers, and deasserts instance bit in DERRLOCd
29-0 —	Reserved

18.8.3.12 Process Identifier (PID0 - PID4)

Offset

Register	Offset
PID0	700h
PID3	70Ch
PID4	710h

Function

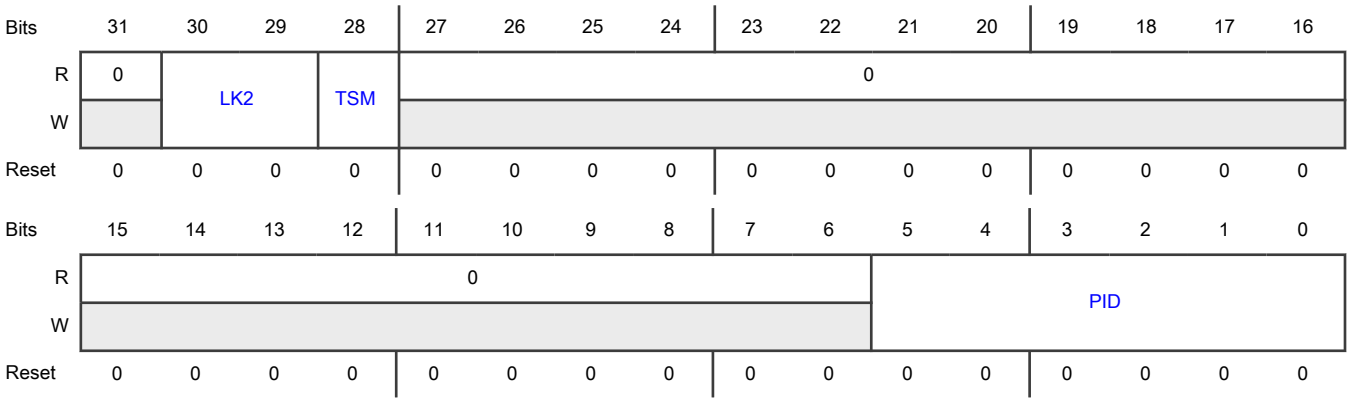
Specifies the PID for the associated core master *m*.

Some cores contain a built-in PID register. If the core has a built-in PID register, XRDC populates the PID field with the value from the core PID register. If the core does not have the built-in register, the XRDC PID register allows applications to mimic PID operation for that core by writing the desired PID to the associated register.

HWCFG2 provides a bitmap of the implemented PID*m* registers. Noncore masters do not have an associated PID register.

For information about PID-based operation, see PID-based domain assignment.

Diagram



Fields

Field	Function
31 —	Reserved
30-29 LK2	Lock Limits or prohibits writes to this register. When you assert a bit in this field, it remains asserted until the next module reset. If the core master has a built-in PID register, as indicated in HWCFG2, then a secure privileged read returns 0 for this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b,01b - Any secure privileged write 10b - Secure privileged writes from master only 11b - Locks
28 TSM	Three-State Model Specifies that the core master supports only the three-state access control model. If you write 1 to this field, it remains asserted until the next reset. For cores that support only the three-state access control model, you must assert this field before loading any nonsecure value into the PID. See Generation of secure attribute.
27-6 —	Reserved
5-0 PID	Process Identifier Specifies the transaction PID for the corresponding core master. If the core has a built-in PID register, then a secure privileged read returns the core's PID register value. Bit 5 specifies the secure attribute (0 = secure, 1 = nonsecure) for the transaction.

18.8.3.13 Master Domain Assignment (MDA_W0_0_DFMT0 - MDA_W0_4_DFMT0)

Offset

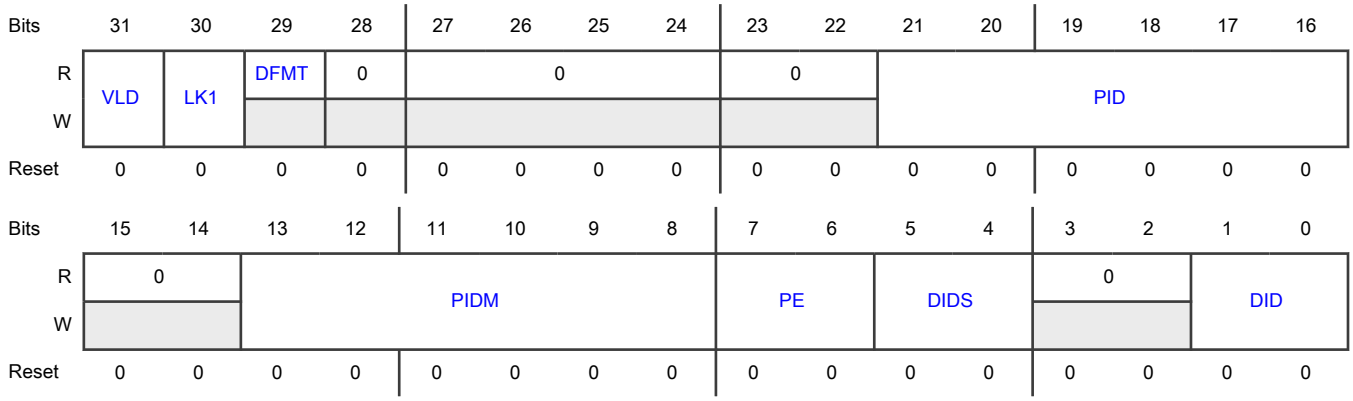
Register	Offset
MDA_W0_0_DFMT0	800h
MDA_W0_3_DFMT0	860h
MDA_W0_4_DFMT0	880h

Function

Specifies the information used by the MDAC to assign a core bus master to a specific domain (DID). For more information, see Master domain assignment controller (MDAC).

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 VLD	Valid Specifies whether this domain assignment is valid. In other words, if VLD and CR[GVLD] are both asserted, XRDC uses the configuration in this register in the domain assignment process. If CR[GVLD] is set to 1 and VLD is set to 0 then every transaction from this master will be assigned a DID of 0. This field has no effect unless XRDC is enabled (CR[GVLD] = 1). 0b - Invalid 1b - Valid
30 LK1	Lock Prohibits writes to this register. • If unlocked, this register accepts any secure privileged write. • If locked, you cannot write to this register and it remains read-only until after the next reset. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Unlocked 1b - Locks When writing 0b - No effect 1b - Locks
29 DFMT	Domain Format Indicates the domain assignment format. 0b - Core bus master domain assignment (DFMT0)
28	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
27-24 —	Reserved
23-22 —	Reserved
21-16 PID	Process Identifier Specifies the PID to be combined with the PIDM field and included in the domain assignment process. This field applies only if PID is enabled (PE = 1).
15-14 —	Reserved
13-8 PIDM	Process Identifier Mask Specifies a mask applied to the PID to support including multiple PIDs in the domain hit determination. For each bit asserted in PIDM, the corresponding bit of the PID is ignored in the comparison. This field applies only if PID is enabled (PE = 1).
7-6 PE	Process Identifier Enable Enables the optional inclusion of PID, qualified by PIDM, in the domain hit evaluation. This inclusion supports the definition of inclusive or exclusive sets of masked PID values. 00b-01b - No PID is included 10b - Partial domain hit = (PID & ~PIDM) == (PIDm[PID] & ~PIDM) 11b - Partial domain hit = ~((PID & ~PIDM) == (PIDm[PID] & ~PIDM))
5-4 DIDS	DID Select Selects the source of the DID. 00b - Use the DID field of this register 01b - Use the input DID 10b - Concatenate bits 3–2 of this register with the least significant 2 bits of the input DID (DID_in[1:0]) 11b - Reserved
3-2 —	Reserved
1-0 DID	Domain Identifier Specifies the DID. DIDS controls whether and how this value is used.

18.8.3.14 Master Domain Assignment (MDA_W0_1_DFMT1 - MDA_W0_5_DFMT1)

Offset

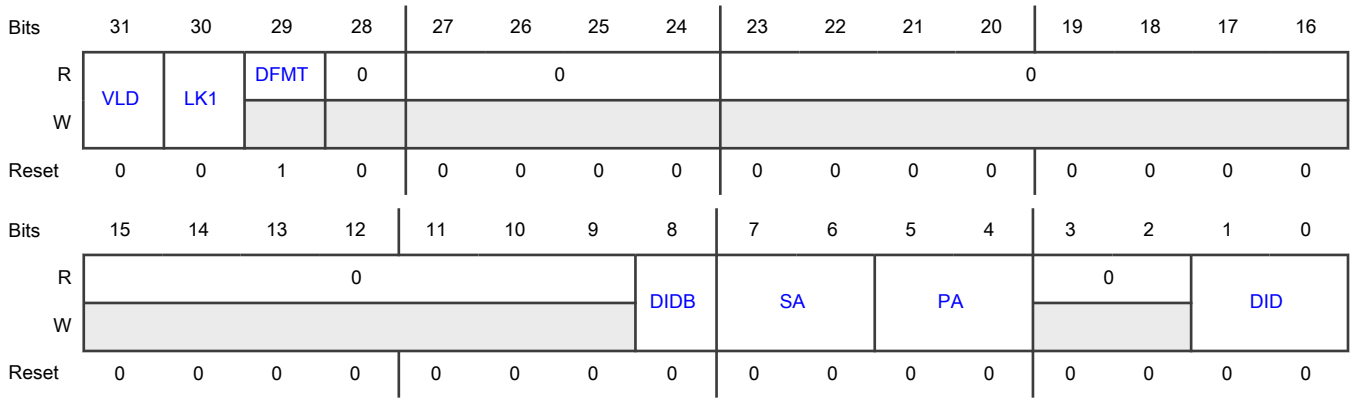
Register	Offset
MDA_W0_1_DFMT1	820h
MDA_W0_2_DFMT1	840h
MDA_W0_5_DFMT1	8A0h

Function

Specifies the information used by the MDAC to assign a bus master to a specific domain (DID). For more information, see Master domain assignment controller (MDAC).

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 VLD	Valid Specifies whether this domain assignment is valid. In other words, if VLD and CR[GVLD] are both asserted, XRDC uses the configuration in this register in the domain assignment process. If CR[GVLD] is set to 1 and VLD is set to 0 then every transaction from this master will be assigned a DID of 0. This field has no effect unless XRDC is enabled (CR[GVLD] = 1). 0b - Invalid 1b - Valid
30 LK1	Lock Prohibits writes to this register. • If unlocked, this register accepts any secure privileged write.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If locked, you cannot write to this register and it remains read-only until after the next reset. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Unlocked 1b - Locks When writing 0b - No effect 1b - Locks
29 DFMT	Domain Format Indicates the domain assignment format. 1b - Bus master domain assignment (DFMT1)
28 —	Reserved
27-24 —	Reserved
23-9 —	Reserved
8 DIDB	DID Bypass Enables bypassing of an input DID as the domain identifier for this master. This capability allows noncore masters (for example, a DMA) to appear as a core. After this field is set to 1, it remains at that value until the next reset. 0b - Bypass DID input. Use DID 1b - Use DID input
7-6 SA	Secure Attribute Specifies the secure attribute. **NOTE** If SA = 1X or VLD = 0, use the secure attribute input from the master. 00b - Force to secure 01b - Force to nonsecure 1xb - Use secure attribute from the master

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-4 PA	Privileged Attribute Specifies the privileged (supervisor/user) attribute. **NOTE** If PA = 1X or VLD = 0, use the privileged attribute input from the master. 00b - Force to user 01b - Force to privileged 1xb - Use privileged attribute from the master
3-2 —	Reserved
1-0 DID	Domain Identifier Specifies the DID.

18.8.3.15 Peripheral Domain Access Control Word 0 (PDAC_W0_32 - PDAC_W0_315)

Offset

Register	Offset
PDAC_W0_32	1100h
PDAC_W0_33	1108h
PDAC_W0_34	1110h
PDAC_W0_35	1118h
PDAC_W0_36	1120h
PDAC_W0_38	1130h
PDAC_W0_39	1138h
PDAC_W0_40	1140h
PDAC_W0_41	1148h
PDAC_W0_42	1150h
PDAC_W0_44	1160h
PDAC_W0_45	1168h
PDAC_W0_46	1170h
PDAC_W0_47	1178h
PDAC_W0_128	1400h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W0_129	1408h
PDAC_W0_130	1410h
PDAC_W0_131	1418h
PDAC_W0_132	1420h
PDAC_W0_133	1428h
PDAC_W0_134	1430h
PDAC_W0_135	1438h
PDAC_W0_136	1440h
PDAC_W0_137	1448h
PDAC_W0_138	1450h
PDAC_W0_139	1458h
PDAC_W0_140	1460h
PDAC_W0_141	1468h
PDAC_W0_142	1470h
PDAC_W0_143	1478h
PDAC_W0_144	1480h
PDAC_W0_145	1488h
PDAC_W0_146	1490h
PDAC_W0_147	1498h
PDAC_W0_148	14A0h
PDAC_W0_149	14A8h
PDAC_W0_150	14B0h
PDAC_W0_151	14B8h
PDAC_W0_152	14C0h
PDAC_W0_153	14C8h
PDAC_W0_154	14D0h
PDAC_W0_155	14D8h
PDAC_W0_156	14E0h
PDAC_W0_157	14E8h
PDAC_W0_158	14F0h
PDAC_W0_159	14F8h
PDAC_W0_160	1500h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W0_161	1508h
PDAC_W0_162	1510h
PDAC_W0_163	1518h
PDAC_W0_164	1520h
PDAC_W0_165	1528h
PDAC_W0_166	1530h
PDAC_W0_167	1538h
PDAC_W0_168	1540h
PDAC_W0_169	1548h
PDAC_W0_170	1550h
PDAC_W0_171	1558h
PDAC_W0_173	1568h
PDAC_W0_175	1578h
PDAC_W0_177	1588h
PDAC_W0_178	1590h
PDAC_W0_179	1598h
PDAC_W0_180	15A0h
PDAC_W0_181	15A8h
PDAC_W0_182	15B0h
PDAC_W0_183	15B8h
PDAC_W0_184	15C0h
PDAC_W0_186	15D0h
PDAC_W0_187	15D8h
PDAC_W0_188	15E0h
PDAC_W0_191	15F8h
PDAC_W0_193	1608h
PDAC_W0_194	1610h
PDAC_W0_195	1618h
PDAC_W0_196	1620h
PDAC_W0_197	1628h
PDAC_W0_198	1630h
PDAC_W0_201	1648h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W0_202	1650h
PDAC_W0_203	1658h
PDAC_W0_204	1660h
PDAC_W0_205	1668h
PDAC_W0_206	1670h
PDAC_W0_207	1678h
PDAC_W0_208	1680h
PDAC_W0_209	1688h
PDAC_W0_212	16A0h
PDAC_W0_213	16A8h
PDAC_W0_214	16B0h
PDAC_W0_215	16B8h
PDAC_W0_216	16C0h
PDAC_W0_217	16C8h
PDAC_W0_219	16D8h
PDAC_W0_220	16E0h
PDAC_W0_221	16E8h
PDAC_W0_223	16F8h
PDAC_W0_224	1700h
PDAC_W0_225	1708h
PDAC_W0_226	1710h
PDAC_W0_227	1718h
PDAC_W0_229	1728h
PDAC_W0_231	1738h
PDAC_W0_232	1740h
PDAC_W0_236	1760h
PDAC_W0_256	1800h
PDAC_W0_257	1808h
PDAC_W0_260	1820h
PDAC_W0_261	1828h
PDAC_W0_262	1830h
PDAC_W0_263	1838h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W0_264	1840h
PDAC_W0_265	1848h
PDAC_W0_266	1850h
PDAC_W0_267	1858h
PDAC_W0_268	1860h
PDAC_W0_269	1868h
PDAC_W0_270	1870h
PDAC_W0_271	1878h
PDAC_W0_272	1880h
PDAC_W0_273	1888h
PDAC_W0_274	1890h
PDAC_W0_275	1898h
PDAC_W0_276	18A0h
PDAC_W0_277	18A8h
PDAC_W0_278	18B0h
PDAC_W0_279	18B8h
PDAC_W0_280	18C0h
PDAC_W0_281	18C8h
PDAC_W0_283	18D8h
PDAC_W0_285	18E8h
PDAC_W0_288	1900h
PDAC_W0_291	1918h
PDAC_W0_292	1920h
PDAC_W0_293	1928h
PDAC_W0_294	1930h
PDAC_W0_295	1938h
PDAC_W0_296	1940h
PDAC_W0_297	1948h
PDAC_W0_298	1950h
PDAC_W0_303	1978h
PDAC_W0_304	1980h
PDAC_W0_307	1998h

Table continues on the next page...

Table continued from the previous page...

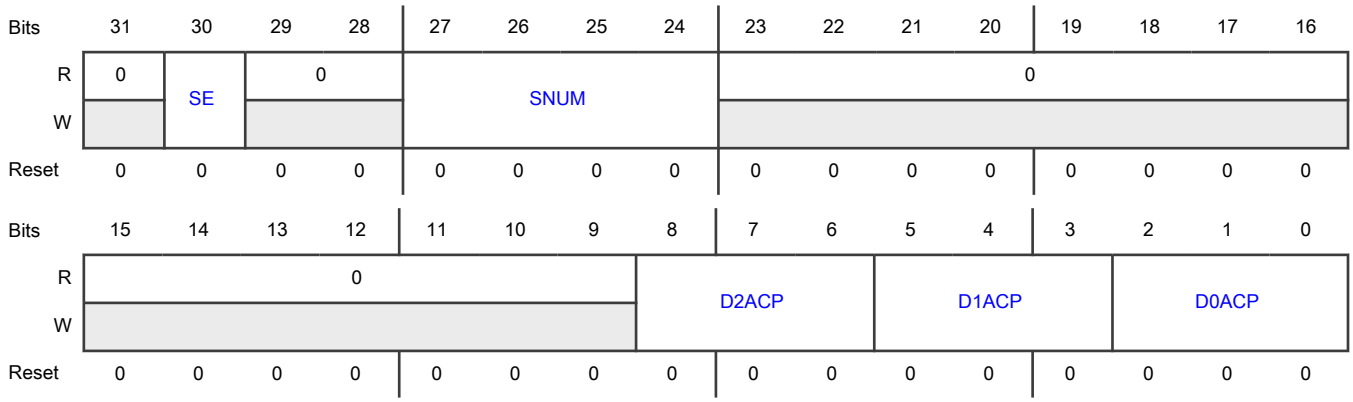
Register	Offset
PDAC_W0_311	19B8h
PDAC_W0_314	19D0h
PDAC_W0_315	19D8h

Function

In conjunction with PDAC_W1_*s*, specifies the ACP configuration for peripheral slot *s*. The ACP controls access to the peripheral by all masters within the domain. For the available ACPs, see Domain ACP specification. For more information, see Peripheral access controller (PAC).

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 —	Reserved
30 SE	Semaphore Enable Enables the inclusion of the semaphore specified in SNUM in the D*d*ACP evaluation. 0b - Disables 1b - Enables
29-28 —	Reserved
27-24 SNUM	Semaphore Number Specifies the hardware semaphore to include in the D*d*ACP access evaluation. This field applies only if you enable semaphore (write 1 to SE).

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-9 —	Reserved
8-6: D2ACP 5-3: D1ACP 2-0: D0ACP	Domain Access Control Policy Specifies the ACP for the associated domain. This field applies only for a supported DID; if the DID is not implemented, the field is read-only zero (no access rights). For field values, see Domain ACP specification.

18.8.3.16 Peripheral Domain Access Control Word 1 (PDAC_W1_32 - PDAC_W1_315)

Offset

Register	Offset
PDAC_W1_32	1104h
PDAC_W1_33	110Ch
PDAC_W1_34	1114h
PDAC_W1_35	111Ch
PDAC_W1_36	1124h
PDAC_W1_38	1134h
PDAC_W1_39	113Ch
PDAC_W1_40	1144h
PDAC_W1_41	114Ch
PDAC_W1_42	1154h
PDAC_W1_44	1164h
PDAC_W1_45	116Ch
PDAC_W1_46	1174h
PDAC_W1_47	117Ch
PDAC_W1_128	1404h
PDAC_W1_129	140Ch
PDAC_W1_130	1414h
PDAC_W1_131	141Ch
PDAC_W1_132	1424h
PDAC_W1_133	142Ch
PDAC_W1_134	1434h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W1_135	143Ch
PDAC_W1_136	1444h
PDAC_W1_137	144Ch
PDAC_W1_138	1454h
PDAC_W1_139	145Ch
PDAC_W1_140	1464h
PDAC_W1_141	146Ch
PDAC_W1_142	1474h
PDAC_W1_143	147Ch
PDAC_W1_144	1484h
PDAC_W1_145	148Ch
PDAC_W1_146	1494h
PDAC_W1_147	149Ch
PDAC_W1_148	14A4h
PDAC_W1_149	14ACh
PDAC_W1_150	14B4h
PDAC_W1_151	14BCh
PDAC_W1_152	14C4h
PDAC_W1_153	14CCh
PDAC_W1_154	14D4h
PDAC_W1_155	14DCh
PDAC_W1_156	14E4h
PDAC_W1_157	14ECh
PDAC_W1_158	14F4h
PDAC_W1_159	14FCh
PDAC_W1_160	1504h
PDAC_W1_161	150Ch
PDAC_W1_162	1514h
PDAC_W1_163	151Ch
PDAC_W1_164	1524h
PDAC_W1_165	152Ch
PDAC_W1_166	1534h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W1_167	153Ch
PDAC_W1_168	1544h
PDAC_W1_169	154Ch
PDAC_W1_170	1554h
PDAC_W1_171	155Ch
PDAC_W1_173	156Ch
PDAC_W1_175	157Ch
PDAC_W1_177	158Ch
PDAC_W1_178	1594h
PDAC_W1_179	159Ch
PDAC_W1_180	15A4h
PDAC_W1_181	15ACh
PDAC_W1_182	15B4h
PDAC_W1_183	15BCh
PDAC_W1_184	15C4h
PDAC_W1_186	15D4h
PDAC_W1_187	15DCh
PDAC_W1_188	15E4h
PDAC_W1_191	15FCh
PDAC_W1_193	160Ch
PDAC_W1_194	1614h
PDAC_W1_195	161Ch
PDAC_W1_196	1624h
PDAC_W1_197	162Ch
PDAC_W1_198	1634h
PDAC_W1_201	164Ch
PDAC_W1_202	1654h
PDAC_W1_203	165Ch
PDAC_W1_204	1664h
PDAC_W1_205	166Ch
PDAC_W1_206	1674h
PDAC_W1_207	167Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W1_208	1684h
PDAC_W1_209	168Ch
PDAC_W1_212	16A4h
PDAC_W1_213	16ACh
PDAC_W1_214	16B4h
PDAC_W1_215	16BCh
PDAC_W1_216	16C4h
PDAC_W1_217	16CCh
PDAC_W1_219	16DCh
PDAC_W1_220	16E4h
PDAC_W1_221	16ECh
PDAC_W1_223	16FCh
PDAC_W1_224	1704h
PDAC_W1_225	170Ch
PDAC_W1_226	1714h
PDAC_W1_227	171Ch
PDAC_W1_229	172Ch
PDAC_W1_231	173Ch
PDAC_W1_232	1744h
PDAC_W1_236	1764h
PDAC_W1_256	1804h
PDAC_W1_257	180Ch
PDAC_W1_260	1824h
PDAC_W1_261	182Ch
PDAC_W1_262	1834h
PDAC_W1_263	183Ch
PDAC_W1_264	1844h
PDAC_W1_265	184Ch
PDAC_W1_266	1854h
PDAC_W1_267	185Ch
PDAC_W1_268	1864h
PDAC_W1_269	186Ch

Table continues on the next page...

Table continued from the previous page...

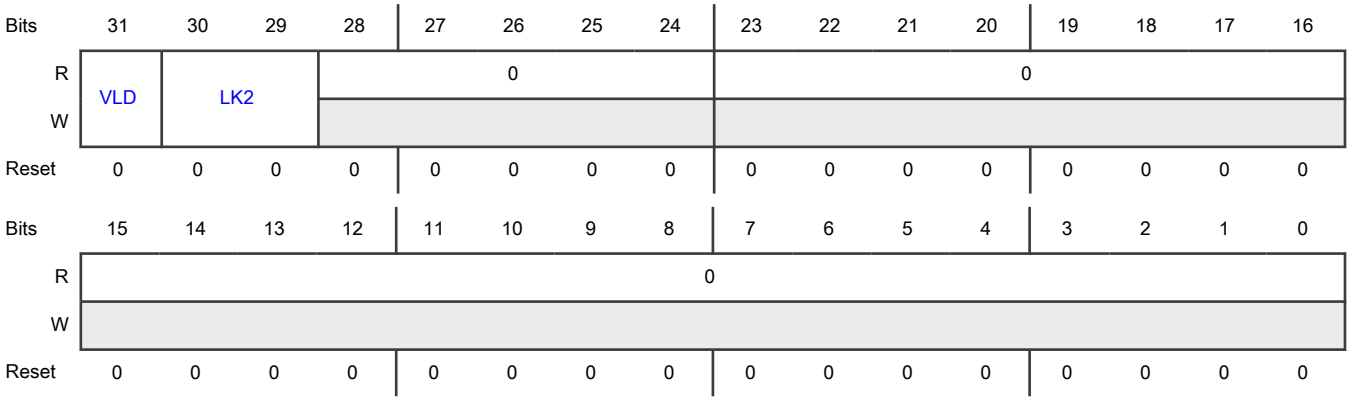
Register	Offset
PDAC_W1_270	1874h
PDAC_W1_271	187Ch
PDAC_W1_272	1884h
PDAC_W1_273	188Ch
PDAC_W1_274	1894h
PDAC_W1_275	189Ch
PDAC_W1_276	18A4h
PDAC_W1_277	18ACh
PDAC_W1_278	18B4h
PDAC_W1_279	18BCh
PDAC_W1_280	18C4h
PDAC_W1_281	18CCh
PDAC_W1_283	18DCh
PDAC_W1_285	18ECh
PDAC_W1_288	1904h
PDAC_W1_291	191Ch
PDAC_W1_292	1924h
PDAC_W1_293	192Ch
PDAC_W1_294	1934h
PDAC_W1_295	193Ch
PDAC_W1_296	1944h
PDAC_W1_297	194Ch
PDAC_W1_298	1954h
PDAC_W1_303	197Ch
PDAC_W1_304	1984h
PDAC_W1_307	199Ch
PDAC_W1_311	19BCh
PDAC_W1_314	19D4h
PDAC_W1_315	19DCh

Function

In conjunction with PDAC_W0_*s*, specifies the ACP configuration for peripheral slot *s*.

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 VLD	Valid Specifies whether this domain assignment is valid. In other words, if VLD and CR[GVLD] are both asserted, XRDC uses the configuration in this pair of registers. If either CR[GVLD] or this field is 0, all accesses to the peripheral are allowed. To support a coherent register state, any write to PDAC_W0_*s* forces this field to zero. This field has no effect unless XRDC is enabled (CR[GVLD] = 1). 0b - Invalid 1b - Valid
30-29 LK2	Lock Limits or prohibits writes to the set of PDAC words (PDAC_W0_*s* and PDAC_W1_*s*) for this peripheral slot. When you assert a bit in this field, it remains asserted until the next module reset. 00b-01b - Both words can be written to 10b - Domain d can update only its associated DdACP field—all other fields are read-only 11b - Locks (both words are read-only)
28-24 —	Reserved
23-0 —	Reserved

18.8.3.17 Memory Region Descriptor Word 0 (MRGD_W0_0 - MRGD_W0_35)

Offset

Registers in this array exist only for the following combinations of index values.

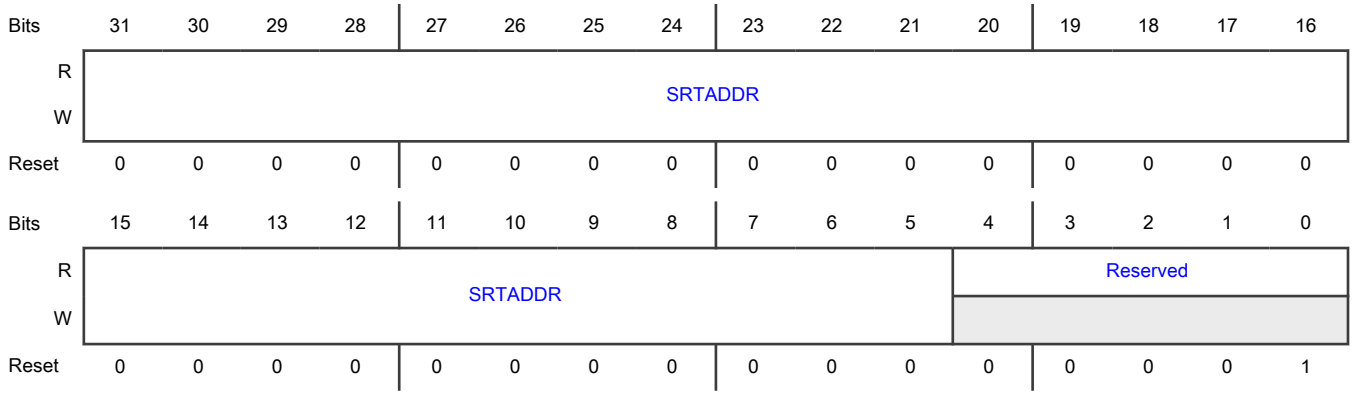
Index *n*	Index *m*
0–1	0–15
2	0–3

Register	Offset
MRGD_W0_(n * 16 + m)	2000h + (n × 200h) + (m × 20h)

Function

Specifies the starting address of memory region *r*.

Diagram



Fields

Field	Function
31-5 SRTADDR	Start Address Specifies the most significant bits of the 0-modulo 32-byte start address of the memory region. The minimum region size is 32 bytes.
4-0 —	Reserved

18.8.3.18 Memory Region Descriptor Word 1 (MRGD_W1_0 - MRGD_W1_35)

Offset

Registers in this array exist only for the following combinations of index values.

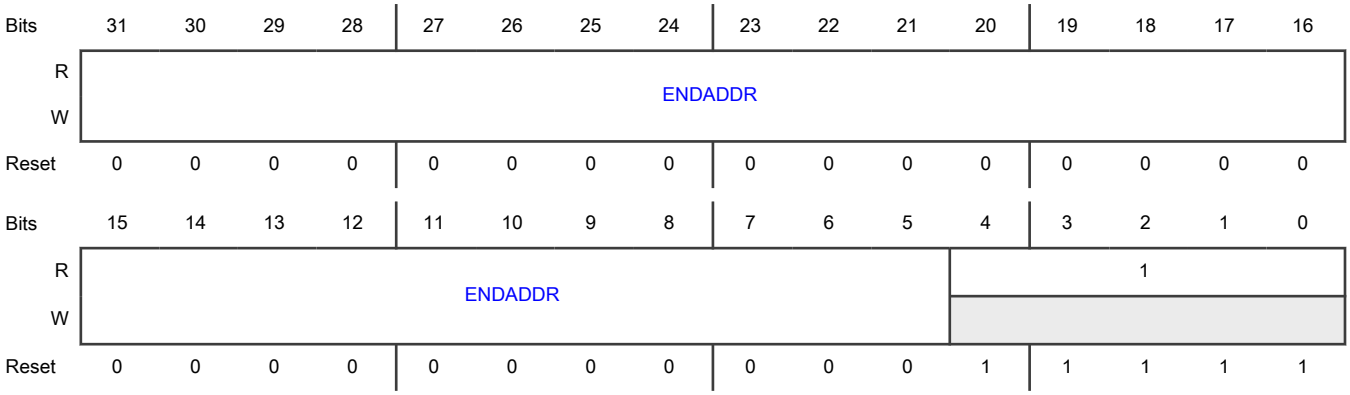
Index *n*	Index *m*
0–1	0–15
2	0–3

Register	Offset
MRGD_W1_(n * 16 + m)	2004h + (n × 200h) + (m × 20h)

Function

Specifies the ending address of memory region *r*.

Diagram



Fields

Field	Function
31-5 ENDADDR	End Address Specifies the most significant bits of the 31-modulo 32-byte end address of memory region *r*.
4-0 —	Reserved

18.8.3.19 Memory Region Descriptor Word 2 (MRGD_W2_0 - MRGD_W2_35)

Offset

Registers in this array exist only for the following combinations of index values.

Index *n*	Index *m*
0–1	0–15
2	0–3

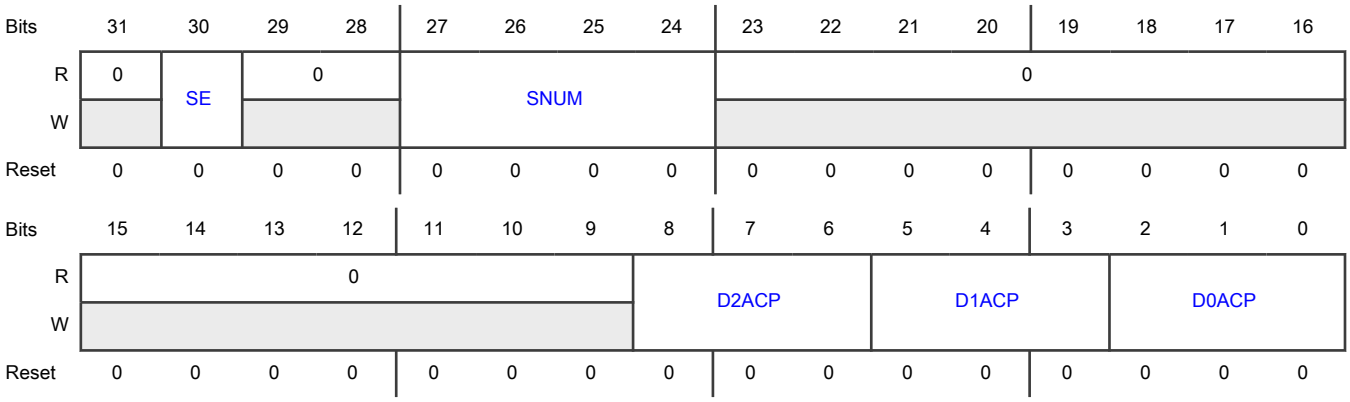
Register	Offset
MRGD_W2_(n * 16 + m)	2008h + (n × 200h) + (m × 20h)

Function

Specifies the ACP for the associated domain. The encodings specify read and write access capabilities based on the four operating states. This field applies only for a supported DID; if the DID is not implemented, the field is read-only zero (no access rights).

For field values, see Domain ACP specification.

Diagram



Fields

Field	Function
31 —	Reserved
30 SE	Semaphore Enable Enables the inclusion of the semaphore specified in SNUM in the D*d*ACP evaluation. 0b - Disables 1b - Enables
29-28 —	Reserved
27-24 SNUM	Semaphore Number Specifies the hardware semaphore to include in the D*d*ACP access evaluation. This field applies only if you enable semaphore (write 1 to SE).
23-9 —	Reserved
8-6: D2ACP 5-3: D1ACP 2-0: D0ACP	Domain Access Control Policy Specifies the ACP for the associated domain. This field applies only for a supported DID; if the DID is not implemented, the field is read-only zero (no access rights). For field values, see Domain ACP specification.

18.8.3.20 Memory Region Descriptor Word 3 (MRGD_W3_0 - MRGD_W3_35)

Offset

Registers in this array exist only for the following combinations of index values.

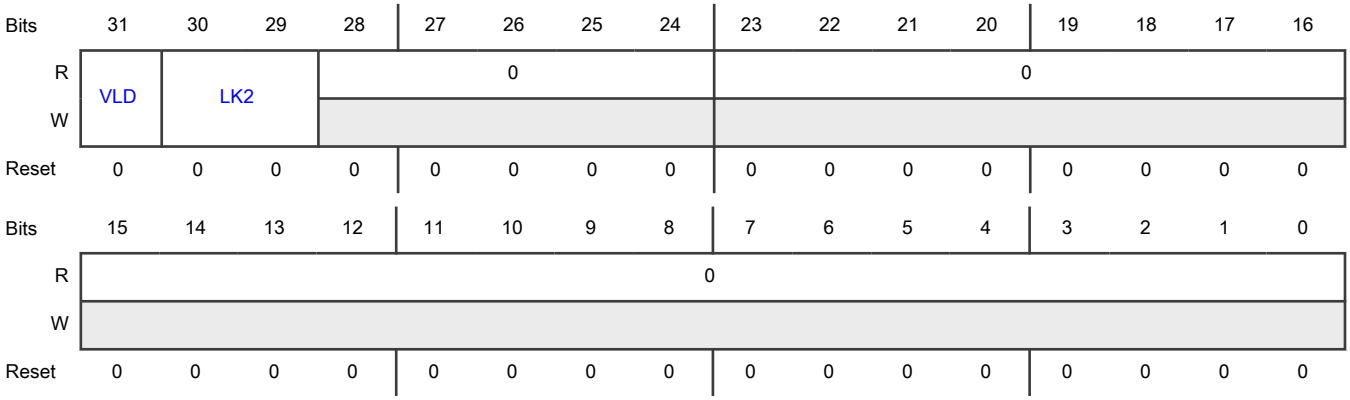
Index *n*	Index *m*
0–1	0–15
2	0–3

Register	Offset
MRGD_W3_(n * 16 + m)	200Ch + (n × 200h) + (m × 20h)

Function

Specifies whether this memory region descriptor is enabled and limits or prohibits writes to it.

Diagram



Fields

Field	Function
31 VLD	Valid Specifies whether this domain assignment is valid. In other words, if VLD and CR[GVLD] are both asserted, XRDC uses the configuration in this set of registers. If CR[GVLD] is 0, all accesses to the memory are allowed. If CR[GVLD] is 1 and VLD is 0, all accesses are blocked. To support a coherent register state, a write to any of the MRGD W0–W2 registers forces this field to zero. This field has no effect unless XRDC is enabled (CR[GVLD] = 1). 0b - Invalid 1b - Valid
30-29	Lock Limits or prohibits writes to the set of MRGD words (MRGD_W*w_n*) for this memory region.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK2	When you assert a bit in this field, it remains asserted until the next module reset. 00b - All words in the set can be written to 01b - Reserved 10b - Domain d can update only its associated DdACP field; all other fields are read-only 11b - Locks (all words are read-only)
28-24 —	Reserved
23-0 —	Reserved

18.9 Glossary

ACP access control policy. The access limitations specified for memory and peripheral resources.

DID domain identifier. A numeric value that identifies a specific domain.

domain An access-controlled virtual group of on-chip masters (cores and noncore masters) and targets (memories and peripherals) that comprise an isolated computing environment. All masters in a domain have the same access to chip resources within that domain.

See Introduction to domains for more information.

LPID logical partition identifier. Also called an operating system ID or VMID, the LPID identifies a virtual master (either core or noncore) that runs on a hypervisor.

master A processor core or non-processor module, such as DMA or a communications channel, that can initiate transactions with memory and peripheral resources.

MDAC Master Domain Assignment Controller. Manages resource assignments and DIDs.

MGR Manager. Manages accesses through the XRDC programming model.

MRC Memory Region Controller. Controls access to memories based on memory region descriptors.

PAC Peripheral Access Controller (also sometimes called PDAC). Controls access to peripherals.

PDAC See PAC.

PID process identifier. A numeric value provided by some core processors to identify the currently active process.

SDAC Deprecated. See MRC.

target A peripheral or memory resource that one or more masters can access. This term replaces "slave."

transaction A read or write request made by a master to a target peripheral or memory.

VMID virtual machine identifier. See LPID.

Chapter 19
Memory and Memory Interfaces

19.1 Introduction

This chapter discusses the configuration of memories and memory interfaces, such as flash memory, flash memory controller, and SRAM.

19.2 Flash memory controller and flash memory modules

For information on this, see the following chapters:

- Flash Memory Controller (PFLASH)
- Embedded Flash Memory

19.3 Related information

Table 81. Related information

Topic	Related chapters	For additional information
System memory map	Memory Map	See the memory map file attached to this document.
Clocking	• Clocking Overview • Clock Generation Module (MC_CGM)	—
Arm Cortex-M7 core	Cortex-M7 Overview	
XRDC	Extended Resource Domain Controller	
Direct-memory access	• Direct Memory Access Multiplexer (DMAMUX) • Enhanced Direct Memory Access (eDMA)	See the DMAMUX map file attached to this document.
EIM	Error Injection Module	—
ERM	Error Reporting Module	—

19.4 SRAM access

In case a master accesses an SRAM with multi-bit ECC errors, the chip may respond as follows:

- Map all such faults to FCCU. The recommended reaction for the fault is to generate the functional reset.
- Map such faults to ERM. If the ERM interrupt is enabled, ERM generates an interrupt.

19.5 Memories

The following table provides information on the types of memories, with their associated configurations, available in MWCT2xxxS family.

Table 82. Memory configuration

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
SRAM0	160 KB	32 KB with STANDBY mode retention (4096 × (64 + 8))	ECC (SECDED)	8	ERM	MWCT2D17S
		128 KB (16384 × (64 + 8))				
	64 KB	32 KB with STANDBY mode retention (4096 × (64 + 8))	ECC (SECDED)	8	ERM	MWCT2D16S
		32 KB (4096 × (64 + 8))				
	96 KB	32 KB with STANDBY mode retention (4096 × (64 + 8))	ECC (SECDED)	8	ERM	MWCT2016S
		64 KB (8192 × (64 + 8))				
	32 KB	32 KB with STANDBY mode retention (4096 × (64 + 8))	ECC (SECDED)	8	ERM	MWCT2015S
	16 KB	16 KB with STANDBY mode retention (2048 × (64 + 8))	ECC (SECDED)	8	ERM	MWCT2014S
SRAM1	160 KB	32 KB (4096 × (64 + 8))	ECC (SECDED)	8	ERM	MWCT2D17S
		128 KB (16384 × (64 + 8))				
CM7_1 I-cache data	8 KB	4 KB (512 × (64 + 8))	ECC (SECDED)	8	ERM	MWCT2D17S MWCT2D16S
		4 KB (512 × (64 + 8))				
CM7_1 I-cache tag	672 bytes	336 bytes (128 × (21 + 7))	ECC (SECDED)	7	ERM	
		336 bytes (128 × (21 + 7))				
CM7_1 D-cache data	8 KB	1 KB (256 × (32 + 7))	ECC (SECDED)	7	ERM	
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
CM7_1 D-cache tag	800 bytes	200 bytes (64 × (25 + 7))	ECC (SECDED)	7	ERM	
		200 bytes (64 × (25 + 7))				
		200 bytes (64 × (25 + 7))				

Table continues on the next page...

Table 82. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
		200 bytes (64 × (25 + 7))				
CM7_0 I-cache data	8 KB	4 KB (512 × (64 + 8))	ECC (SECDED)	8	ERM	MWCT2D17S MWCT2016S
		4 KB (512 × (64 + 8))				
CM7_0 I-cache tag	672 bytes	336 bytes (128 × (21 + 7))	ECC (SECDED)	7	ERM	MWCT2015S MWCT2D16S
		336 bytes (128 × (21 + 7))				
CM7_0 D-cache data	8 KB	1 KB (256 × (32 + 7))	ECC (SECDED)	7	ERM	
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
CM7_0 D-cache tag	800 bytes	200 bytes (64 × (25 + 7))	ECC (SECDED)	7	ERM	
		200 bytes (64 × (25 + 7))				
		200 bytes (64 × (25 + 7))				
		200 bytes (64 × (25 + 7))				
DMA TCD	1 KB	1 KB (128 × (64 + 8))	ECC (SECDED)	8	ERM	MWCT2D17S
	640 bytes	640 bytes (80 × (64 + 8))	ECC (SECDED)	8	ERM	MWCT2016S MWCT2015S
FlexCAN_0	5 KB	5 KB (640 × (64 + 40))	ECC (SECDED)	40	FlexCAN_0	MWCT2D17S
FlexCAN_0	3968 bytes	3968 bytes (496 × (64 + 40))	ECC (SECDED)	40	FlexCAN_0	MWCT2015S, MWCT2016S, MWCT2D16S
FlexCAN_1	1920 bytes	1920 bytes (240 × (64 + 40))	ECC (SECDED)	40	FlexCAN_1	MWCT2D17S MWCT2016S
FlexCAN_2	1920 bytes	1920 bytes (240 × (64 + 40))	ECC (SECDED)	40	FlexCAN_2	MWCT2015S MWCT2D16S
FlexCAN_3	1152 bytes	1152 bytes (144 × (64 + 40))	ECC (SECDED)	40	FlexCAN_3	MWCT2D17S MWCT2016S MWCT2D16S

Table continues on the next page...

Table 82. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
FlexCAN_4	1152 bytes	1152 bytes (144 × (64 + 40))	ECC(SECDED)	40	FlexCAN_4	MWCT2D17S MWCT2016S
FlexCAN_5	1152 bytes	1152 bytes (144 × (64 + 40))	ECC (SECDED)	40	FlexCAN_5	
EMAC TX	8960 bytes	4480 bytes (1024 × (35 + 7))	ECC (SECDED)	7	EMAC	MWCT2D17S, MWCT2D16S
		4480 bytes (1024 × (35 + 7))				
EMAC RX	8960 bytes	4480 bytes (1024 × (35 + 7))	ECC (SECDED)	7	EMAC	
		4480 bytes (1024 × (35 + 7))				
EMAC TSN	1664 bytes	1664 bytes (512 × (26 + 7))	ECC (SECDED)	7	EMAC	
EMAC RXPARSER	960 bytes	960 bytes (80 × (96 + 8))	ECC (SECDED)	8	EMAC	
QuadSPI TX	320 bytes	320 bytes (80×32)	No	0	Not applicable	
Cortex-M7 cluster ETF ETMI	1 KB	1 KB (128×64)	No	0	Not applicable	MWCT2D17S
Cortex-M7 cluster ETF ETMD	2 KB	2 KB (128×128)	No	0	Not applicable	
HTM ETF	1 KB	1 KB (128×64)	No	0	Not applicable	
Shared system ETF	2 KB	2 KB (256×64)	No	0	Not applicable	
HSE secure RAM	64 KB	64 KB (8192 × (64 + 8))	ECC (SECDED)	8	HSE	MWCT2D17S, MWCT2016S, MWCT2015S, MWCT2D16S
HSE PKC	8 KB	4 KB (512 × (64 + 8))	ECC (SECDED)	8	HSE	
		4 KB (512 × (64 + 8))				
HSE MTB	1 KB	1 KB (256×32)	No	0	HSE	
CM7_0_ITCM	32kB	4096x(64+8)	ECC (SECDED)	8	ERM	MWCT2015S, MWCT2016S, MWCT2D16S, MWCT2D17S
CM7_0_DTCM	64kB	8192x(32+8)	ECC (SECDED)	8	ERM	MWCT2015S, MWCT2016S, MWCT2D16S, MWCT2D17S
		8192x(32+8)				

Table continues on the next page...

Table 82. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
CM7_1_ITCM	32kB	4096x(64+8)	ECC (SECDED)	8	ERM	MWCT2D16S, MWCT2D17S
CM7_1_DTCM	64kB	8192x(32+8)	ECC (SECDED)	8	ERM	MWCT2D16S, MWCT2D17S
		8192x(32+8)				

19.6 Recommendations for ARM memories

As per ARM M-7 Safety manual, following considerations must be ensured for proper operation of ARM memories:

- ITCM and DTCM must be properly initialized with correct ECC before any read operation to avoid any code runaway or software malfunction or core lockup.

NOTE

ITCM must be initialized with 64-bit writes whereas DTCM can be initialized with 32-bit writes also.

- To safely disable TCM:
  1. Clear ITCMCR[EN] or DTCMCR[EN] as required. See ARM Cortex-M7 Devices Generic User Guide for details on ITCMCR and DTCMCR register.
  2. Execute DSB instruction
  3. Execute ISB instruction

NOTE

Care must be taken if disabling the ITCM while executing from it. In this case, software must ensure that the switch-off code is stored in the L2 code memory region from where execution continues when the ITCM is disabled.

- To safely disable the I-cache:
  1. Clear CCR[IC]. See ARM Cortex-M7 Devices Generic User Guide for details on CCR register.
  2. Execute DSB instruction
  3. Execute ISB instruction
- To safely disable the D-cache:
  1. Clean and invalidate non-WT locations in D-cache
  2. Clear CCR[DC]
  3. Execute DSB instruction

See Table 83 for details on memory ECC initialization.

19.7 Memory ECC initialization summary

The table below summarizes memory ECC initialization.

Table 83. Memory ECC initialization summary

Memory	Write access size	Masters

Table continues on the next page...

Table 83. Memory ECC initialization summary (continued)

		CM7_ n	CM7_ m	eDMA
SRAM	64-bits only	System	System	System
CM7_n ITCM	64-bits only	Direct or Backdoor	Backdoor	Not possible
CM7_n DTCM	32-bits or 64-bits	Direct or Backdoor	Backdoor	Backdoor
CM7_m ITCM	64-bits only	Backdoor	Direct or Backdoor	Not possible
CM7_m DTCM	32-bits or 64-bits	Backdoor	Direct or Backdoor	Backdoor

19.8 Glossary

DTCM	Data tightly coupled memory
ECC	Error code correction
ETF	Embedded trace FIFO
ETMD	Embedded trace macrocell-data
ETMI	Embedded trace macrocell-instruction
ITCM	Instruction tightly coupled memory
MTB	Macrocell trace buffer
PKC	Public key cryptography
RXPARSER	Receive parser memory
SECDED	Single error correction double error detection
SRAM	Static random access memory
TSN	Time sensitive network

Chapter 20
Embedded Flash Memory (c40asf)

20.1 Chip-specific c40asf flash memory information

20.1.1 Flash memory configuration and register settings

Table 84 shows flash memory blocks and their associated configuration.

Table 84. Flash block configuration

Block or address number	Block name	Start address (hex)	End address (hex)	Size	Applicability
0	Code flash memory 0	0040_0000	0047_FFFF	512 KB	MWCT2014S, MWCT2015S
		0040_0000	004F_FFFF	1 MB	MWCT2D17S, MWCT2D16S, MWCT2016S
1	Code flash memory 1	0048_0000	004F_FFFF	512 KB	MWCT2015S
		0050_000	005F_FFFF	1 MB	MWCT2D17S, MWCT2D16S, MWCT2016S
2	Code flash memory 2	0060_0000	006F_FFFF	1 MB	MWCT2D17S
3	Code flash memory 3	0070_0000	007F_FFFF	1 MB	MWCT2D17S
4[1]	Data flash memory	1000_0000	1001_FFFF	128 KB	MWCT2D17S
2[1]				128 KB	MWCT2D16S, MWCT2016S
		1000_0000	1000_FFFF	64 KB	MWCT2015S, MWCT2016S
0 [2]	UTest NVM	1B00_0000	1B00_1FFF	8 KB	MWCT2015S, MWCT2016S, MWCT2D16S, MWCT2D17S, MWCT2014S

1. Sector operation should be used for data flash during high voltage operation.
2. The address region number is the same as block number for all the blocks except this one. Address region is called UTest NVM address region in this case.

No application cores except HSE_B can access the address regions shown in Table 85.

Table 85. Secure flash memory configuration

Description	MWCT2015S		MWCT2016S		MWCT2D17S		Size
	Start address	End address	Start address	End address	Start address	End address	
Secure BAF code (SBAF_CODE)	004F_4000h	004F_FFFFh	005F_4000h	005F_FFFFh	007F_4000h	007F_FFFFh	48 KB

NOTE

For HSE memory configuration, see *'Memory map when HSE firmware usage feature flag is enabled'* section in *'Boot Overview'* chapter.

The flash memory can perform multiple reads in parallel (between different blocks), using the single-, dual-, or quad-read feature. It also includes an internal UTest mode that can generate single- and double-bit ECC errors.

The code flash memory lets you access data paths as per the following table:

Chip	No. of data paths that can be accessed simulataneously
MWCT2D17S	3
MWCT2016S	2

You can configure the flash memory sectors as erase- or write-protected by using the flash memory's sector and super-sector locking features.

Data flash and/or Code flash memory blocks can be altered by a code executing from different flash block or SRAM.

Data flash memory is the same as code flash memory—supporting program, erase, and read operations.

After the completion of the program or erase operations, the MCR and MCRS registers notify you as soon as the code flash memory code can be executed. At the end of reset recovery, MCRS[DONE] transitions from 0 to 1.

NOTE

MCRS[RWE] might become 1 during the program or erase operation because of speculative accesses by Cortex-M7 core(s). If there are memory regions where no speculative accesses must be initiated, Arm recommends to configure the on-core MPU to set those regions with these attributes:

- Device or strongly-ordered
- Execute never

NOTE

- ADR[A3] and ADR[A4] field are reserved for MWCT2016S, as code flash memory size in 2 MB for MWCT2016S.
- For MWCT2016S, reset value of FLASH_SSPELOCK and FLASH_XSSPELOCK register is 0x0FFF_FFFF.

20.1.2 MCRE register reset value

The reset value of the MCRE register depends on the chip-specific flash memory configuration. This table provides the flash memory configuration of the chip and the associated register reset values.

Table 86. MCRE register reset value

MCRE fields	Reset value
n1M (bit position 23:21)	100b – Four 1 MB blocks
	010b – Two 1 MB blocks
n2M (bit position 31:29)	100b – Four 2 MB blocks
n512K (bit position 15:14)	00b – Zero 512 KB blocks
	01b – One 512 KB block
	10b – Two 512 KB blocks
n256K (bit position 7:6)[1]	01b – One 256 KB blocks
MCRE register reset value	MWCT2014S - 0000_8040h
	MWCT2015S - 0000_4040h

Table continues on the next page...

Table 86. MCRE register reset value (continued)

MCRE fields	Reset value
	MWCT2D17S - 0080_0040h
	MWCT2x16S - 0040_0040h

1. For block size less than equal to 256 KB

20.1.3 Debugger considerations while Flash Program/Erase

While flash programming is done via debugger, special considerations should be taken care off. See "Debug Subsystem" chapter for details.

20.2 Introduction

The primary function of the embedded flash memory serves as an electrically programmable and erasable non-volatile memory (NVM) that may be used for instruction or data storage.

20.2.1 Overview

The embedded flash memory is designed for use in embedded MCU and SoC applications. It supports a total memory size up to 8.5 MB of NVM in main space and 8 KB of UTest NVM space per module. Multiple flash modules may be instantiated within an SoC to achieve higher density. The embedded flash memory is addressable by page (256 bits) for read operation and double-word(s) and page and quad page for program operations. Flash memory reads always return 256 bits, although read page buffering may be performed by the PFC. The flash memory is able to do multiple reads in parallel (between different blocks), utilizing the quad read feature.

For details on the embedded flash memory architecture and features, see Functional description.

The following figure shows the top-level diagram and functional organization of the flash memory unit.

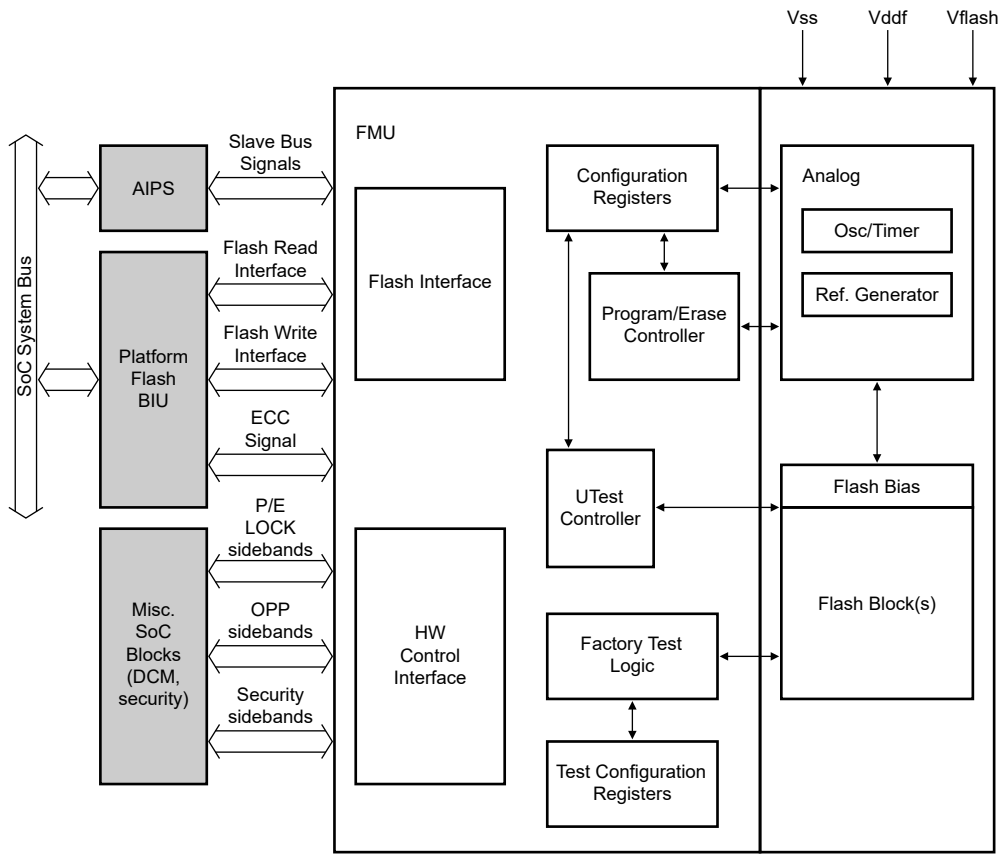


Figure 48. Block diagram

20.2.2 Features

The embedded flash memory includes these distinct features:

- Test information stored in a dedicated non-volatile sector (referred to as the UTest sector)
- OTP space made available in the UTest sector
- Read page size of 256 bits (8 words)
- ECC with single bit correction, double bit detection (all 1s valid) with 64-bit granularity
- Quad page programming (256-bit granularity)
- Hardware programmable sector and super sector program or erase restriction control
- Erasing of selected sector or block erase
- Independent programming of the UTest NVM sector
- Embedded hardware program and erase algorithms
- Support for reading while writing when the accesses are to different blocks
- UTest mode (user-accessible test modes) including array integrity and margin read
- Triple-voted flops for safety-critical flash memory functions (for example, internal trimming, redundancy, mode control, and others)

20.2.3 Modes of operation

Following is a brief description of the embedded flash memory operating modes.

- User mode is the default operating mode of the embedded flash memory. In this mode, it is possible to read and write registers (including register-based interlock writes), read the memory array, program the memory array, and erase the memory array. In this mode program and erase operations are initiated by doing register writes. Program and erase operations are controlled by an internal state machine.
- Low-Power mode turns off all DC current sources within the embedded flash memory and enables VFLASH and VDDF to be power gated. The embedded flash memory is not accessible for read, write, program, or erase when in Low-Power mode. This mode results in the lowest current draw that embedded flash memory can achieve.
- UTest mode is a tiered test mode strategy in which a portion of the factory test modes are made available. This mode is protected but accessible.

20.3 UTest NVM sector

The UTest NVM sector may be enabled by PEADR[PEASAD] during interlock writes. When the UTest NVM space is enabled, all program and erase operations are mapped to the UTest NVM sector. User-mode programming of the UTest NVM sector is enabled only when PEADR[PEASAD] is high.

The UTest NVM sector is an OTP sector (assuming Test mode disable seal is written) . Therefore, performing an erase operation is not permitted.

The UTest NVM sector supports RWW, and is grouped with the sectors in partition 0.

The UTest NVM sector may be locked against program by using hardware program and erase protection input to the flash module (MCRS[TSPELOCK]).

The UTest NVM sector contains specified data that is needed for embedded flash memory or SoC features.

Programming of the UTest NVM space has restrictions that are similar to those of the main space in terms of how ECC is calculated. Only one program operation is allowed per 64-bit ECC segment.

20.4 Test mode disable seal

The UTest NVM sector includes a mechanism to disable factory entry into Test mode selectable by block. Extreme care must be taken when using this feature, because blocks that are selected to be protected in this method will not have possible failures analyzed by factory failure analysts. Once sealed, the UTest sector becomes OTP (erase locked, and OPP enabled). The UTest sector is not accessible in Test Mode.

Protection of this sort prevents all high voltage operations to the flash memory executed by the internal state-machine, as well as reads through this state machine, and reads through the array integrity state machine when using Test mode interfaces.

UTest operations, margin read and array integrity, are also protected by preventing MISR updates on blocks selected for protection, although reads are still performed, and single bit corrections and double bit detections will be logged. Protection through other interfaces is provided by normal User mode protection mechanisms and is device-specific.

The method to disable factory entry into Test mode is to first program the Test mode disable seal location to be 5A4B_3C2Dh. After the next reset is asserted, Test mode is disabled.

It is possible to create a password to enable factory entry into Test mode. This can be programmed into the Test mode disable override passcode. The passcode may not be 0000_0000h, FFFF_FFFFh, or 5555_5555h. These are all invalid passcodes and will not be accepted to override. If it is desired, by the customer, that override never be possible, one of the three invalid passcodes must be put into the Test mode disable override passcode location. Passcode may be entered to authenticate entry, if enabled, by performing a 32-bit register write to register address 90h.

Even if the Test mode disable seal password is written, the UTest NVM sector is always protected and the UTest operations, margin read and array integrity, are always protected.

Only blocks selected in the Test mode disable block select field are controlled by the Test mode disable feature. Therefore, it is possible for customers to selectively pick blocks that have this type of protection and will not be eligible for factory failure

analysis. Bits programmed to 0 in the Test mode disable block select field(s) designate blocks that are controlled by the Test mode disable feature.

In order to let two opportunities to select blocks for the Test mode disable, two regions are available, Test mode disable bock select - 1 and Test mode disable block select - 2. If either Test mode disable block select - 1 or Test mode disable block select - 2 has a 0 programmed, that block will be protected. The bits of these two regions are logically ANDed to define the blocks that are effectively selected for the Test mode disable feature. The bock select field is organized as shown in the next table:

Table 87. Test mode disable block select

Block	Bits used to select blocks
Block 0 disable	Data[0]
Block 1 disable	Data[8]
Block 2 disable	Data[16]
Block 3 disable	Data[24]
Block 4 disable	Data[32]
Block 5 disable	Data[40]

20.5 Functional description

The embedded flash memory module consists of blocks with the following options:

- 2 MB blocks (0, 1, 2, 3 or 4 blocks allowed per module)
- 1 MB blocks (0, 1, 2, 3 or 4 blocks allowed per module)
- 512 KB blocks (0, 1, 2 or 4 blocks allowed per module)
- 256 KB blocks (0, 1, 2 or 4 blocks allowed per module)
- A total of six blocks are allowed per module

Read while Write is allowed within a module, and is determined based on block boundaries. Read while Write partitions are used to determine locations for valid read-while-write (RWW) operations. While the embedded flash memory performs a write (program or erase) to a given partition, it can simultaneously perform a read from any other partition. For program and erase operations, only the address specified by an interlock write determines the partition being written (sector or super sector locking does not determine the RWW partitions being written).

The main address space is also divided into sectors and super sectors to implement independent erase and program protection. The UTest NVM space also exists as a sector and has independent program protection. The UTest NVM sector is included to support systems that require non-volatile memory (NVM) for security or to store system initialization information.

A number of MCR bits are protected against write when another field, or set of fields, is in a specific state. These write locks are covered on a bit-by-bit basis in Module Configuration Register. The write locks do not consider the effects of trying to write two or more bits simultaneously. This module does not allow fields to be written simultaneously that put the device into an illegal state. This is implemented through a priority mechanism among the fields, and the next table shows this mechanism.

Table 88. MCR field set and clear priority levels

Priority level	MCR field(s)
1	ERS
2	PGM
3	EHV

If two or more MCR fields are written simultaneously, only the field with the lowest priority number is accepted for modification. For example, writing 1 to ERS and PGM simultaneously results in 1 being written only to ERS.

Each read of the embedded flash memory retrieves a page, or eight consecutive words (256 bits), of information. The address for each word retrieved within a page differs from the other addresses in the page only by address bits [4:2]. The flash memory page read architecture easily supports both cache and burst mode at the PFC BIU level for high-speed read applications.

The embedded flash memory supports fault tolerance through error correction code (ECC) and error detection. ECC implemented within the embedded flash memory corrects single bit failures and detects double bit failures.

Program and erase of the embedded flash memory requires multiple system clock cycles to complete. Program and erase may be aborted.

The embedded flash memory may operate in various modes as described in the following sections.

20.5.1 User mode

In User mode, the embedded flash memory may be read and written (register writes and interlock writes), programmed or erased. The following sub-sections define all actions that may be performed in User mode.

20.5.1.1 Read and write

The default state of the embedded flash memory is read. The main and UTest NVM address spaces can be read only in the read state. The flash registers are always available for read, except when the embedded flash memory is in Low-Power mode. The module enters the read state on reset, and remains in the read state under two sets of conditions:

- The read state is active when the embedded flash memory is enabled (User mode read).
- The read state is active when MCR[PGM] or MCR[ERS] are 1 and a high voltage operation is ongoing (RWW).

NOTE

Reads done to the partition(s) being operated on (either erased or programmed) result in MCRS[RWE] becoming 1.

In embedded flash memory, flash core reads return 256 bits (1 page). Register reads return 32 bits of data.

NOTE

Flash core reads are performed through the PFC BIU. In many cases, the BIU does read page buffering to allow sequential reads to be done with higher performance. This could provide a data coherency issue that must be handled with software. Data coherency may be an issue after a program or erase operation, as well as UTest NVM sector operations.

In user mode, registers may be written (including interlock writes using registers).

Register reads to unmapped register address space return all 0s.

Register writes to unmapped register address space have no effect.

Interlock writes that are attempted during a high voltage operation (MCR[EHV] = 1 or MCRS[DONE] =0) result in the interlock write being ignored, and address and data will not be updated.

20.5.1.2 Program

A flash memory program sequence operates on any page within the flash core. Within a page, up to eight words may be altered in a single program operation. Also, up to four pages can be altered in a single program operation. Whenever the array is programmed, the ECC bits also get programmed.

ECC is handled on a 64-bit boundary. Thus, if only one word in any given 64-bit ECC segment is programmed, the adjoining word in that segment should not be programmed, because ECC calculation is already complete for that 64-bit segment. Attempts to program the adjoining word may likely result in an operation failure. It is recommended that all programming operations ranger from 64 bits to 1024 bits, and be 64-bit aligned. The programming operation should completely fill selected ECC segments within the page. Only one program is allowed per 64-bit ECC segment between erases.

Caution

In rare cases, over-programming of a 64-bit ECC segment may be done (EEPROM emulation in data flash region). In this case, approved EEPROM emulation drivers must be used, and they must limit the number of over-program operations to three times (four total programs between erases).

Programming changes the value stored in an array bit from logic 1 to logic 0 only. Programming cannot change a stored logic 0 to a logic 1.

NOTE

If a logic 0 is attempted to be over-programmed by a logic 1, the resulting operation fails (MCRS[PEG] = 0), and the 0s that are interlocked are merged (ORed) with the 0s that are already present in the 64-bit ECC segment, unless the block is designated as an over-orogram protected block.

Addresses in locked sectors cannot be programmed. Values may be programmed in any or all of eight words, within a page, with a single program sequence. Up to four pages can be programmed at once on a quad-page boundary. The program operation consists of the following sequence of events:

1. Write the address to be programmed using logical address registers located in the PFC. The result of this write will also be reflected physically in the PEADR register.

   NOTE

   PEADR writes are initiated by writes that first occur in the PFC. Please see the platform flash memory controller chapter for more information. PEADR reads are allowed, and represent the flash memory physical address.

   NOTE

   Ensure the sector that contains the address to be programmed is unlocked. Sector and super sector lock status is latched at this step and does not change until the next program or erase sequence is started.

2. Data to be programmed must be written in the appropriate DATAX register (where X is 0 to 31). All unwritten data words default to FFFF_FFFFh.
3. Change the value in MCR[PGM] from a 0 to a 1.
4. Write a 1 to MCR[EHV] to start the internal program sequence or skip to step 8 to terminate.
5. Wait until MCRS[DONE] becomes 1.

   NOTE

   Since MCRS[DONE] clears with MCR[EHV] being set, it may not be possible for software to read MCRS[DONE] as 0 between step 4 and step 5, depending on the operation selected.

6. Confirm MCRS[PEG] = 1.
7. Write 0 to MCR[EHV].
8. Write 0 to MCR[PGM] to terminate the program sequence.

Program may be initiated with the writing of the PEADR register (enabled with a write done in the PFC block). The PEADR write to initiate the program sequence determines the quad-page address to be programmed. Data must also be written to the DATA register(s). This first write is referred to as an interlock write. Unwritten locations default to a data value of FFFF_FFFFh.

An interlock write must be performed before writing 1 to MCR[PGM], and MCR[PGM] must be set before writing 1 to MCR[EHV] during a program sequence. A program sequence may be terminated by writing 0 to MCR[PGM] prior to writing 1 to MCR[EHV].

While MCR[EHV] is 1 and MCRS[DONE] is 0, MCR[EHV] may be cleared, resulting in a program abort. A program abort forces the embedded flash memory to step 8 of the program sequence. An aborted program results in PEG becoming 0, indicating a failed operation. The data space being operated on before the abort contains indeterminate data.

Caution

Aborting a program operation leaves the FC addresses being programmed in an indeterminate data state. This may be recovered by executing an erase on the affected block.

20.5.1.2.1 Program hardware locking

Hardware locking which affects the program and erase operations is available for UTest sector and all array blocks.

20.5.1.2.2 Over-program protection enable

One-time programming can be enabled. If over-program protection is enabled, any double word that has already been programmed cannot be programmed again. The over-program protection enable does not affect the erase operation. Attempts to over-program result in MCRS[PEG] becoming 0. If any double word within a quad-page has an over-program protection violation, MCRS[PEG] becomes 0, and no double words are programmed.

One-time programming can be enabled for the UTest NVM sector.

20.5.1.3 Erase

An erase changes the value stored in all bits of the selected sector or block to logic 1, and operates on a sector or a block in the main address space. The erase sequence is fully automated within the flash memory. Locked sectors cannot be erased. The erase sequence consists of the following events:

1. Write to the sector or block address to be erased using logical address registers located in the PFC. The result of this write is reflected physically in the PEADR register. One, and only one, DATA register must also be written. This is referred to as an erase interlock write.

   NOTE

   PEADR writes are initiated by writes that first occur in the PFC. Please see the platform memory flash controller chapter for more information. PEADR reads are allowed, and represent the flash memory physical address.

   NOTE

   The selected block or sector must be unlocked prior to initiating an erase with the appropriate sector or super sector lock registers. Sector and super sector lock status is latched at this step and does not change until the next program or erase sequence starts. If a block is selected for erase, all the sectors and super sectors must be unlocked within that block.

2. Change the value in MCR[ERS] from a 0 to a 1, and at the same time select the size of erase using MCR[ESS].
3. Write a 1 to MCR[EHV] to start the internal erase sequence or skip to step 6 to terminate.
4. Wait until MCRS[DONE] becomes 1.

   NOTE

   Since MCRS[DONE] clears with MCR[EHV] being set, it may not be possible for software to read MCRS[DONE] as 0 between step 3 and step 4, depending on the operation selected.

5. Confirm MCRS[PEG] = 1.
6. Write 0 to MCR[EHV].
7. Write 0 to MCR[ERS] to terminate the erase, and if set MCR[ESS] should also be cleared (else the field auto clears).

Erase may be initiated with the writing of the PEADR register (enabled with a write done in the Platform Flash Controller block). The PEADR write to initiate the erase sequence determines the sector or block to be erased. One write must also occur to a DATA register. Data word written during an erase sequence interlock write is ignored. This write is referred to as an erase interlock write.

An erase interlock write must occur before writing 1 to MCR[ERS] (and optionally MCR[ESS]), and MCR[ERS] must be set before writing 1 to MCR[EHV] during an erase sequence. The erase sequence may be terminated by writing 0 to MCR[ERS] prior to writing 1 to MCR[EHV].

While MCR[EHV] is 1 and MCRS[DONE] is 0, MCR[EHV] may be cleared, resulting in an erase abort. An erase abort forces the embedded flash memory to step 7 of the erase sequence. An aborted erase results in PEG becoming 0, indicating a failed operation. The sector or block being operated on before the abort contain indeterminate data.

20.5.1.3.1 Erase hardware locking

Hardware locking which affects the erase and program operations is available for UTest sector and all array blocks. For details, see Program hardware locking.

20.5.1.4 Express program

A mechanism is made available to provide an express method to program up to 1024 bits of flash memory. When activated, all ongoing program and erase operations through the register interface are automatically aborted (MCR[EHV] becomes 0 automatically) and the express interface is given priority. When using the express interface, sequence requirements are ignored, although a XPEADR write is required to start the express program sequence.

The flash express program mechanism is provides a quicker way to get information programmed into a block of flash memory when an application needs an immediate diary access, or a crash log access.

Addresses in locked sectors cannot be express programmed.

The express program operation consists of the following sequence of events:

1. Write the address to be programmed using the logical address register located in the PFC. The result of this write will also be reflected physically in the XPEADR register.

   NOTE

   XPEADR writes are initiated by writes that first occur to the PFC. Please see the platform flash controller chapter for more information. XPEADR reads are allowed, and represent the flash memory physical address.

   NOTE

   Ensure the sector that contains the address to be programmed is unlocked. Sector and super sector lock status is latched at this step and does not change until the next program, erase or express program is started.

   NOTE

   If XMCR[XPGM] interrupts a program or erase setup after MCR[EHV] is written to 1, the MCR and MCRS registers indicate if the operation was automatically aborted (MCR[EHV] = 0, MCRS[PEG] = 0, MCR[PGM] =1 or MCR[ERS] = 1), or if the operation was successful (MCR[EHV] = 1, MCRS[PEG] = 1, MCR[PGM] =1 or MCR[ERS] = 1). If express program interrupts a program or erase setup prior to MCR[EHV] being written to a 1, the MCR and PEADR registers will be preserved to show the state of the setup upon interruption. If fields MCR[PGM] = 1 or MCR[ERS] = 1 at interruption, the user must terminate the event by writing 0 to these fields. However, if MCR[PGM] or MR[ERS] were not 1 at interruption, PEADR is preserved and does not need to be rewritten, but the PDATA registers are required to be rewritten for the interrupted operation. After the PDATA write(s) are repeated to finish the interlock write, the remainder of the program and erase sequence can be done to restart the interrupted operation (including doing a terminate after PGM or ERS are written to 1, if desired).

   NOTE

   Before starting the express program, ensure that UTest operations (array integrity or user margin read) are not ongoing (or about to be started). UT0[UTE] must be low before requesting the express program, and remain low during express program operations. Hardware locks are in place to ensure this occurs.

2. Wait until XMCR[XDOK] goes high.
3. Write the DATA registers with the data desired to be programmed. Up to 1024b (128B) may be programmed with express program. Only DATA registers that are written will receive program pulses.
4. Change the value in XMCR[XPGM] from a 0 to a 1.
5. Write a 1 to XMCR[XEHV] to start the internal express program sequence.

6. Wait until XMCR[XDONE] becomes 1.
7. Confirm XMCR[XPEG] = 1.
8. Write 0 to XMCR[XEHV].
9. Write 0 to XMCR[XPGM] to terminate the express program sequence.

20.5.2 Low-Power mode

In Low-Power mode, the embedded flash memory is put into a state where VDDF power gating and VFLASH power gating is allowed. It is the lowest current mode for the flash memory. No reads from or writes to the embedded flash memory are possible when in this mode. Most power dissipation is due to leakage in this mode.

Prior to entering Low-Power mode, it is required that all program, erase, and UTest operations be stopped.

When in Low-Power mode, register access is prevented. Flash core accesses are also prevented until power mode is exited. Flash core reads and writes may occur after power mode is exited.

The embedded flash memory returns to a post-reset state in all cases.

20.5.3 Program and erase watchdog timer

The embedded flash memory contains an internal watchdog timer that is used to prevent denial of service issues and events.

The watchdog timer is active during two periods of a program and erase event: program/erase setup and program/erase cleanup. If it is desired that program and erase operations also be checked with a watchdog timer, a system resource may be used for this purpose.

Program/erase setup always starts at the time of the PEADR write. An express program setup starts with a write to XPEADR and XMCR[XDOK] = 1. In the event of an Express program interrupt of main or alternate interface setup, the timer starts at the time when the express program operation completes (write of XMCR[XPGM] = 0). The watchdog stops when either MCR[EHV] is set to start the operation, or when the operation is terminated. Termination can occur when MCR[PGM] or MCR[ERS] are written to 0 (prior to MCR[EHV] being written to 1), or MCRS[PES] written to 0. If the express program interrupts the program/erase setup, this also stops the timer on that interface, and express program maintains priority. The above statements also apply to the alternate interface.

NOTE

If PEADR writes occur while UT0[UTE] is 1, the watchdog feature does not activate for setup. Watchdog timer is not active during UTest operations.

Program/erase cleanup always starts at the time of MCRS[DONE] = 1 on completion or an abort. In the event of an express program auto-abort of main or alternate interface, the cleanup starts at the time of MCRS[DONE] becoming 1 and the express program operation completed (XMCR[XPGM] written to 0). The watchdog stops when MCR[PGM] or MCR[ERS] are written to 0 at the end of the operation. If express program interrupts the program/erase cleanup, the timer on that interface also stops. The above statements also apply to the alternate interface.

Watchdog timers exist for each interface (main, alternate and express).

If a watchdog timeout occurs, the watchdog interrupt will be reflected in MCRS[WDI], AMCRS[AWDI] or XMCR[XWDI]. Interrupts can be enabled on the main and alternate interfaces using MCR[WDIE] or AMCR[AWDIE]. When a watchdog timeout occurs and an interrupt register asserts, the PEID restrictions on that interface are released. At that point, any master can complete the operation, and PEID locking is no longer enforced. If the watchdog timeout occurs, the EHV bit on that interface (MCR[EHV], AMCR[AEHV], and XMCR[XEHV]) are locked for writing to a 1. In this instance, any master can take the interface to a program and erase terminate. This enables accepted MCRS[WDI], AMCRS[AWDI] or XMCR[XWDI] to become 0, and the operation may restart with a new interlock write.

NOTE

If the watchdog timeout occurs before PGM becomes 1 (on any interface), or ERS becomes 1 (on any interface), the sequence must be taken to the point that the PGM or ERS bit for that interface is written and then the field for that interface can be cleared to create the terminate event and to clear the watchdog interrupt.

By default, the watchdog is enabled. Through the alternate MCR, the watchdog may be disabled (AMCR[WDD]), and the watchdog timeouts maybe adjusted (AMCR[WDT]) for all timeouts and all interfaces.

20.5.4 UTest mode

UTest mode enables customers to do specific tests that check the integrity of the embedded flash memory.

NOTE

When entering UTest mode, a best practice is to ensure all error flags and address error reporting in ADR are cleared to their reset value to ensure robust software and procedure execution.

NOTE

UTest mode diagnostic features ECC Logic Check, EDC after ECC Logic Check, Address Encode Logic Check, and Read Reference and Voltage Check, require serialization of flash reads starting with mode setup and ending with mode exit. This may be accomplished by executing diagnostic code from system RAM (with single reads to flash tagged address) as well as ensuring any flash reads outside of this diagnostic code are idle (single core).

20.5.4.1 Array integrity self check

Array integrity is checked using a predefined address sequence (based on UT0[AIS]), and this operation is executed on selected blocks.

The data to be read is customer-specific user code programmed into the flash memory and the correct MISR signature is calculated based on that code.

Any random or nonrandom code is valid. After the operation completes, the results of the reads can be checked by reading the MISR value, to determine if an incorrect read or ECC detection was noted. Array integrity MISR value is calculated after ECC detection and correction. Array integrity requires that the read wait states control registers in the CTL register be set to match the system frequency being used.

The array integrity self check consists of the following steps:

1. Enable UTest mode.
2. Select the block to receive the array integrity check by performing an interlock write to that block. Write logical address register in the PFC (which will be reflected in the PEADR register) and 1 DATA register (PDATA is ignored). The block selected for array integrity check does not need to be unlocked.

   NOTE

   Blocks protected with the Test mode disable seal are still read as part of the array integrity sequence. The resulting read on sealed blocks is not captured in the MISR, but single bit correction, double bit detection and breakpoints are still honored.

   NOTE

   It is not possible to perform array integrity operations on the UTest NVM sector.

3. If desired, write 1 to UT0[AIS] for sequential addressing only.

   NOTE

   For normal integrity checks of the flash memory, sequential addressing is recommended. If it is required to more thoroughly check the read path (in a diagnostic mode), AIS shall remain 0 to examine more read transitions. This sequence takes more time.

4. Seed the MISR registers (UM0 - UM9) with desired values.
5. If breakpoints are desired, write 1 to UT0[AIBPE], and ensure that MCRS[EER] and MCRS[SBC] are 0. If it is desired to break on a single bit correction, ensure that UT0[SBCE] = 1.
6. Write 1 to UT0[AIE].

a. If desired, the array integrity operation may be aborted before UT0[AID] goes high. This may be done by writing 0 to UT0[AIE] and then continuing to the next step. Note that in the event of an aborted array integrity check, MISRs contain a signature for the portion of the operation that was completed prior to the abort and are not deterministic. Before performing another array integrity operation, the UM0, UM1, UM2, UM3, UM4, UM5, UM6, UM7, UM8, and UM9 registers may need to be initialized to the desired seed value by doing register writes.

b. If desired, the array integrity operation may be suspended before UT0[AID] goes high. This may be done by writing 1 to UT0[AISUS] to request an array integrity suspend. After UT0[AISUS] becomes 1, the user should wait for UT0[AID] to become 1, which indicates that the flash memory has entered the suspend state, and normal reads to the flash memory may be done. After UT0[AID] becomes 1, UT0[AISUS] may be written to 0 to resume the array integrity sequence.

NOTE

User mode array reads requested during the array integrity test are ignored to ensure that the array integrity operation is not corrupted. The memory array does not respond to array read requests during this time. User mode array reads may be executed if suspended or at a breakpoint.

7. Wait until UT0[AID] becomes 1.
8. If breakpoints are enabled, check if UT0[NAIBP] = 1. If the value of this field is 1, MCRS[EER], MCRS[SBC], and ADDR may be checked to determine the cause of the break and the address of the break. Prior to resuming the operation, write 1 to clear MCRS[SBC] or MCRS[EER]. Then, the operation may be resumed by writing 0 to UT0[NAIBP]. Continue to wait until UT0[AID] becomes 1. If breakpoints are not enabled, or if UT0[NAIBP] = 0 when UT0[AID] becomes 1, the operation is complete. Continue to the next step.
9. Read values in MISRs (UM0 - UM9) to ensure correct signature.

NOTE

Array integrity reads may be done to unselected (or non-present) locations. Reads done to these locations do not update the MISRs, MCRS[EER] and MCRS[SBC].

10. Write 0 to UT0[AIE].

20.5.4.2 User margin read

User margin read may be done using the array integrity interface, and has all the associated features of the array integrity interface (MISR and breakpoints).

User margin reads are done at a read margin level, checking for erased bits or programmed bits encroaching on the nominal read level.

User margin read requires that the Read Wait States control registers in the CTL register be written to match the system frequency being used. Margin ECC corrections and detections are noted during the user margin read test. Margin read MISR value is calculated after ECC detection and correction.

The data to be read is customer-specific user code programmed into the flash memory.

Any random or non-random code is valid. After the operation completes, the margin read results can be checked by reading MCRS[EER] and the MCRS[SBC] to determine if zero, one, or two bits are being detected by the margin read, as well as checking the MISR signature.

The use model for margin read is in the event of a user-detected single bit correction (through user reads), a margin read may be done to check for a possible second bit falling within the selected margin levels.

The user margin read consists of the following steps:

1. Enable UTest mode.
2. Select the block to receive the array integrity check by performing an interlock write to that block. Write logical address register in the PFC (which will be reflected in the PEADR register) and 1 DATA register (PDATA is ignored). The block selected for user margin read check does not need to be unlocked.

NOTE

Blocks protected with the Test mode disable seal are still read as part of the margin read sequence. The resulting read on sealed blocks is not captured in the MISR, but single bit correction, double bit detection and breakpoints are still honored.

NOTE

It is not possible to perform margin read operations on the UTest NVM sector.

3. Write 1 to UT0[AIS] for sequential addressing only.

NOTE

For margin read checks of the flash memory, sequential addressing is recommended. Writing 0 to UT0[AIS] is possible for margin reads, but using the sequence takes more time, and is not recommended.

4. Seed the MISR registers (UM0 - UM9) with desired values.
5. Ensure that MCRS[EER] and MCRS[SBC] are 0.
6. To enable single bit correction reporting during margin read, write UT0[SBCE] = 1.
7. Write 1 to UT0[MRE].
8. Write UT0[MRV] to the desired value depending on if it is desired to do one's margin or zero's margin.
9. Write 1 to UT0[AIE].

NOTE

User mode array reads requested during the margin read test are ignored to ensure that the margin read operation is not corrupted. The memory array does not respond to array read requests during this time. User mode array reads may be executed if suspended or at a breakpoint.

NOTE

During margin read operations, with UT0[AID] = 0, it is not recommended to attempt write operations to MCRS[SBC] and MCRS[EER]. It is recommended that these fields only be written during suspend, breakpoints or at the completion of the margin read operation.

10. Wait until UT0[AID] becomes 1.
11. If breakpoints are enabled or a suspend was requested during margin read, the operation may be at a breakpoint or a suspend state. See Array integrity self check for more information.
12. Read values in the MISRs (UM0 - UM9) to ensure correct signature.

NOTE

Margin reads may be done to unselected (or non-present) locations. Reads done to these locations do not update the MISRs, MCRS[EER] and MCRS[SBC].

13. Write 0 to UT0[AIE].

20.5.4.3 ECC logic check

ECC logic can be checked by providing data to be read in UD0[EDATA], UD1[EDATA] and/or UD2[EDATAC]. Array reads can then be performed, ensuring expected results.

The ECC logic check consists of the following steps:

1. Enable UTest mode.
2. Write 1 to UT0[EIE].
3. Write to UD0[EDATA], UD1[EDATA] and/or UD2[EDATAC] to provide data and check bit values to be read. Single bit detections or double bit corrections can be simulated by properly selecting data and check bit combinations.

4. Write the page address to receive data provided in step 3 into ADR.
5. Write to UD2[ED3], UD2[ED2], UD2[ED1] and/or UD2[ED0] to select the double words on the page to receive the check.
6. Reads can now be done through the BIU using an array read request. In the event of a BIU read requested from an address that matches the address in ADR, expected data and corrections or detections are observed based on data written into UD0[EDATA], UD1[EDATA] and/or UD2[EDATAC]. MCRS[EER] and MCRS[SBC] can be checked to evaluate the status of reads done.

NOTE

In the event of an ECC error or single bit correction, during the ECC logic check (UT0[EIE] = 1), ADR is not loaded, and the address tagged to receive the ECC logic check values is preserved.

7. Once completed, write UT0[EIE] to 0.

20.5.4.4 EDC after ECC logic check

EDC after ECC logic can be checked by providing data to be read in UD3[EDDATA], UD4[EDDATA] and/or UD5[EDDATAC]. Array reads can then be performed, ensuring expected results.

The EDC after ECC logic check consists of the following steps:

1. Enable UTest mode.
2. Write 1 to UT0[EDIE].
3. Write to UD3[EDDATA], UD4[EDDATA] and/or UD5[EDDATAC] to provide data and check bit values to be read.
4. Write the page address to receive data provided in step 3 into ADR.
5. Write to UD2[EDD3], UD2[EDD2], UD2[EDD1] and/or UD2[EDD0] to select the double words on the page to receive the check.
6. Reads can now be done through the BIU using an array read request. In the event of a BIU read requested from an address that matches the address in ADR, expected EDC after ECC errors are observed based on data written into UD3[EDDATA], UD4[EDDATA] and/or UD5[EDDATAC]. MCRS[EEE] can be checked to evaluate the status of reads done.
7. Once completed, write UT0[EDIE] to 0.

20.5.4.5 Address encode logic check

Address encode logic can be checked by inverting the address encode information from the memory array in UA0[AEI] and UA1[AEI]. Array reads can then be performed, ensuring expected results.

The Address encode check consists of the following steps:

1. Enable UTest mode.
2. Write 1 to UT0[AEIE].
3. Write to UA0[AEI] and/or UA1[AEI] to provide address bit(s) to be inverted.
4. Write the page address to receive inverted addresses provided in step 3 into ADR.
5. Reads can now be done through the BIU using an array read request. In the event of a BIU read requested from an address that matches the address in ADR, expected address encode errors are observed based on address invert values written into UA0[AEI] and/or UA1[AEI]. MCRS[AEE] can be checked to evaluate the status of reads done.
6. Once completed, write UT0[AEIE] to 0.

20.5.4.6 Read reference and voltage check

Read reference and voltage detection logic can be checked by writing 1 to UT0[RRIE]. Array reads can be then be performed, ensuring expected results.

The read reference and voltage check consists of the following steps:

1. Enable UTest mode.
2. Write 1 to UT0[RRIE].
3. Write the page address to receive a read reference error MCRS[RRE] and a read voltage error MCRS[RVE] into ADR.
4. Reads can now be done through the BIU using an array read request. In the event of a BIU read requested from an address that matches the address in ADR, expected read reference and read voltage errors are observed based on ADR registers. MCRS[RRE] and MCRS[RVE] can be checked to evaluate the status of reads done.
5. Once completed, write UT0[RRIE] to 0.

20.5.5 Data flash memory requirements for EEPROM emulation

The embedded flash memory may be used to emulate an EEPROM utilizing software drivers, strategic over-programming of double words in the flash memory and following the below requirement and best practices in EEPROM software drivers.

It is required that the EEPROM emulation driver must build in fault tolerance allowing for the ability to “skip” records. In the case of an unsuccessful program, the ability to retry programming in the next available record location is required (record retirement). In the case of an unsuccessful sector erase, the ability to retire a sector is required (sector retirement).

In addition, following are best practices for the EEPROM emulation software driver to be considered:

- Choose a record scheme which allows for grouping of EEPROM data contents that are updated at one time such that unchanged data is not needlessly copied while minimizing record qualifier and status overhead. A variable length record scheme may be used to allow for grouping of data that is written together and limiting total record overhead.
- Copy only the valid records during a sector change and re-constitution of EEPROM data set to avoid needless copying of data within flash memory. This avoids rebuilding the entire data set.
- Only redundantly storing critical data which cannot be recovered in two sectors if attempting to protect against record or sector failures.
- Load level total program/erase cycles applied to flash memory locations allocated for EEPROM emulation by using a round-robin sector scheme.

When combining round-robin with sector retirement requirement, three or more sectors must be allocated for EEPROM emulation inclusive of extra spare sector(s).

For details, see NXP application notes published on this topic.

20.6 Initialization information

A reset is the highest priority operation for the embedded flash memory and terminates all other operations.

The embedded flash memory uses reset to initialize register and status fields to their default reset values. If the embedded flash memory is executing a program or erase operation (MCR[PGM] = 1 or MCR[ERS] = 1) and a reset is issued, the operation aborts and the embedded flash memory disables the high voltage logic without causing any damage to the high voltage circuits. Reset aborts all operations and forces the embedded flash memory into User mode, ready to receive accesses.

After reset is requested, MCRS[DONE] becomes 0, and remains low during reset and reset recovery.

At the end of reset recovery, MCRS[DONE] transitions from 0 to 1.

After a reset completes, register reads may be performed.

NOTE

Registers that require updating from UTest NVM information, or other inputs, may not read updated values until MCRS[DONE] becomes 1.

During reset recovery, register writes are not allowed until MCRS[DONE] becomes 1 to indicate reset recovery is complete.

Caution

Resetting during a program or erase operation leaves the FC blocks being programmed or erased in an indeterminate data state. This may be recovered by executing an erase on the affected blocks.

20.7 Memory map and register description

20.7.1 c40asf flash memory register descriptions

The embedded flash memory map consists of a flash memory array (which includes main array space and UTest NVM space) and a region of registers associated with the programming model that enable flash memory array operation and modification.

The address space consists of up to 6 blocks (with restrictions), and blocks can be 2 MB, 1 MB, 512 KB or 256KB in size.

20.7.1.1 c40asf_flash memory map

FLASH base address: 402E_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration (MCR)	32	RW	00FF_0000h
4h	Module Configuration Status (MCRS)	32	RW	0000_C100h
8h	Extended Module Configuration (MCRE)	32	R	0000_0000h
Ch	Module Control (CTL)	32	RW	0000_0600h
10h	Address (ADR)	32	RW	0000_0000h
14h	Program and Erase Address (PEADR)	32	R	0000_0000h
50h	Sector Program and Erase Hardware Lock (SPELOCK)	32	R	FFFF_FFFFh
54h	Super Sector Program and Erase Hardware Lock (SSPELOCK)	32	R	0000_0FFFh
70h	Express Sector Program and Erase Hardware Lock (XSPELOCK)	32	R	FFFF_FFFFh
74h	Express Super Sector Program and Erase Hardware Lock (XSSPELOCK)	32	R	0000_0FFFh
90h	Test Mode Disable Password Check (TMD)	32	RW	0000_0000h
94h	UTest 0 (UT0)	32	RW	0000_0001h
98h - B8h	UMISRn (UM0 - UM8)	32	RW	0000_0000h
BCh	UMISR9 (UM9)	32	RW	0000_0000h
D0h	UTest Data 0 (UD0)	32	RW	0000_0000h
D4h	UTest Data 1 (UD1)	32	RW	0000_0000h
D8h	UTest Data 2 (UD2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
DCh	UTest Data 3 (UD3)	32	RW	0000_0000h
E0h	UTest Data 4 (UD4)	32	RW	0000_0000h
E4h	UTest Data 5 (UD5)	32	RW	0000_0000h
E8h	UTest Address 0 (UA0)	32	RW	0000_0000h
ECh	UTest Address 1 (UA1)	32	RW	0000_0000h
F0h	Express Module Configuration (XMCR)	32	RW	00FF_C000h
F4h	Express Program Address (XPEADR)	32	R	0000_0000h
100h - 17Ch	Program Data (DATA0 - DATA31)	32	RW	FFFF_FFFFh

20.7.1.2 Module Configuration (MCR)

Offset

Register	Offset
MCR	0h

Function

NOTE

- A number of Module Configuration Register (MCR) bits are locked against write by other bits. These locks are discussed in relationship to each bit in this section. Simultaneously writing bits which lock each other out is also discussed in Functional description.
- See Functional description for information about simultaneous MCR writes, and priority levels.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PECIE	0		WDIE	0			PGM	0		ESS	ERS	0			EHV
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 PEID	Program and Erase Master/Domain ID This field shows the ID of the master that has started the Program or Erase sequence (as well as Array Integrity and User Margin Reads). The ID is latched when the sequence has started (writing of the PEADR register). Upon completion of an operation (Program (MCR[PGM] written to 0 and MCRS[PES] equals 0), Erase (MCR[ERS] written to 0 and MCRS[PES] equals 0), Array Integrity (UT0[AIE] written to 0), User Margin Read(UT0[AIE] written to 0)), or Program and Erase Sequence clear (MCRS[PES] cleared to 0) the PEID field will return to an all ones state. See the chip-specific information for a list of Master IDs.
15 PECIE	Program/Erase Complete Interrupt Enable PECIE provides a mechanism to trigger an interrupt request upon the assertion of the DONE status due to a high voltage event (program or erase) finishing (normal, abort). If PECIE is written while not in a high voltage event, the interrupt will not immediately trigger, but will trigger after the next high voltage event is completed. Writing (and reading) of this bit will be restricted to the master that updated the PEADR register as captured in the PEID register at the start of the Program or Erase sequence. **NOTE** See the chip-specific information for interrupt details 0b - Interrupt request not generated when MCRS[DONE] is 1 1b - Interrupt request generated when MCRS[DONE] is 1
14-13 —	Reserved
12 WDIE	Watch Dog Interrupt Enable WDIE provides a mechanism to trigger an interrupt request upon the assertion of the WDI bit on the main interface. If WDIE is asserted, when WDI asserts, an interrupt from the flash will trigger to the system. The interrupt from the flash will mirror exactly the WDI bit. WDIE does not affect the register bit WDI. Writing (and reading) of this bit will be restricted to the master that updated the PEADR register as captured in the PEID register at the start of the Program or Erase sequence. **NOTE** See the chip-specific information for interrupt details 0b - Watchdog interrupt not enabled 1b - Watchdog interrupt enabled
11-9 —	Reserved
8 PGM	Program

Table continues on the next page...

Table continued from the previous page...

Field	Function
	PGM is used as part of the setup for a program operation. A 0 to 1 transition of PGM after program interlock write(s) is part of the program sequence. A 1 to 0 transition of PGM ends the program sequence. PGM can be set while in user mode read (ERS is low and UTE is low), and after the interlock write is completed. PES must also be low to enable PGM to be set. PGM can be cleared only when EHV is low and DONE is high. PGM is cleared on reset. Writing (and reading) of this bit will be restricted to the master that updated the PEADR register as captured in the PEID register at the start of the Program or Erase sequence. 0b - Flash memory not executing a program sequence 1b - Flash memory executing a program sequence
7-6 —	Reserved
5 ESS	Erase Size Select ESS is used to qualify the embedded flash memory erase operation for either sector erase or block erase. If ESS is selected for block erase, only the main array of that block will be erased. **NOTE** Block erase is for Factory use only, under controlled conditions. See Appendix A (Electrical Specifications) for more information on environmental condition requirements, and cycle limit. **NOTE** If interlock write occurs to UTest sector, ESS will be locked from setting to a 1. Block Erase of UTest sector is not supported, only sector erase is allowed. Erase of UTest space is not allowed if sealed, although ERS can be set and a PEG error will occur in response to the HV request. ESS can only be written to a 1 at the same time that ERS is written to a 1. If ESS is set, ESS will auto clear with ERS clearing (independent of the value written to ESS). ESS can not be cleared by writing a zero. ESS is cleared on reset. Writing (and reading) of this bit will be restricted to the master that updated the PEADR register as captured in the PEID register at the start of the Program or Erase sequence. 0b - Flash memory erase is on a sector 1b - Flash memory erase is on a block
4 ERS	Erase ERS is used as part of the setup for an erase operation. A 0 to 1 transition of ERS after an erase interlock write is part of the erase sequence. A 1 to 0 transition of ERS ends the erase sequence. ERS can only be set in user mode read (PGM is low and UTE is low), and after the interlock write is completed. PES must also be low to enable ERS to be set. ERS can be cleared only when EHV is low and DONE is high. ERS is cleared on reset. Writing (and reading) of this bit will be restricted to the master that updated the PEADR register as captured in the PEID register at the start of the Program or Erase sequence. 0b - Flash memory not executing an erase sequence

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Flash memory executing an erase sequence
3-1 —	Reserved
0 EHV	Enable High Voltage The EHV bit enables the embedded flash memory for a high voltage program/erase operation. EHV is cleared on reset. EHV must be set after the PGM or ERS bit is set in a program or erase sequence. Writing (and reading) of this bit will be restricted to the master that updated the PEADR register as captured in the PEID register at the start of the Program or Erase sequence. EHV may be set, initiating a program/erase under one of the following conditions: • Erase (ERS = 1, PEP = 0, PES = 0, WDI = 0) • Program (PGM = 1, PEP = 0, PES = 0, WDI = 0) Clearing EHV while DONE is low will abort the current program/erase high voltage operation. An abort causes the value of PEG to be cleared, indicating a failed program/erase; address locations being operated on by the aborted operation contain indeterminate data after an abort. EHV may not be written to a 1 after an abort is requested (EHV being cleared) and before DONE transitions high. EHV may not be written to a 1 after it is cleared as part of a program or erase cleanup, until after the next PEADR write. **NOTE** Aborting a high voltage operation leaves addresses in an indeterminate data state. This may be recovered by executing an erase on the affected sectors. 0b - Flash memory is not enabled to perform a high voltage operation. 1b - Flash memory is enabled to perform a high voltage operation.

20.7.1.3 Module Configuration Status (MCRS)

Offset

Register	Offset
MCRS	4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EER	SBC	AEE	EEE	0		RVE	RRE	0			RWE	0		PEP	PES
W	W1C	W1C	W1C	W1C			W1C	W1C				W1C			W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DONE	PEG	0	WDI	0		EPEG	TSPELOCK	0							RE
W																
Reset	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EER	ECC Event Error This bit provides information on previous reads (either user initiated reads or internally initiated reset reads).If a double bit detection occurred, the EER bit is set to a '1'. This bit must then be cleared, or a reset must occur before it returns to a 0 state. This bit may not be set by software. In the event of a single bit detection and correction, this bit is not set. If EER is not set, or remains 0, this indicates that all previous reads (from the last reset, or clearing of EER) are correct. Since this bit is an error flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads occurring normally 1b - ECC error occurred during a previous read
30 SBC	Single Bit Correction SBC provides information on previous reads, if the SBCE is set. If a single bit correction occurred, the SBC bit is set to a 1. This bit must then be cleared, or a reset must occur before this bit returns to a 0 state. If SBC is not set, or remains 0, this indicates that all previous reads (from the last reset, or clearing of SBC) did not require a correction. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads occurring without corrections 1b - Single bit correction occurred during a previous read
29 AEE	Address Encode Error AEE provides information on previous reads monitoring the address encode feature. On every read request to the flash, the incoming address is compared to an encoded address (row, column, and block) coming back from the memory array using the read data sense amplifier timing. If these two values do not match (zero selected, multiple selected, wrong selected), or the timing is incorrect, an address encode error will be recorded. If an address encode mismatch is detected, this bit is set to indicate that previous reads requested may have been corrupted. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads are occurring without address encode mismatches 1b - Previous read may be corrupted based on address encode mismatch

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 EEE	EDC after ECC Error EEE provides information on previous reads monitoring the EDC after ECC feature. On every read request to the flash, ECC is recalculated serially, and if there is a mismatch between the ECC calculations (taking into account corrections or detections) a late error will be reported. If an EDC after ECC mismatch is detected, this bit is set to indicate that previous reads requested may have been corrupted. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads are occurring without EDC after ECC mismatches 1b - Previous read may be corrupted based on ECC calculation errors
27-26 —	Reserved
25 RVE	Read Voltage Error RVE provides information on previous reads monitoring the read voltage. If the read voltage is detected to be out of range, this bit is set to indicate that previous reads requested may have been corrupted. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads are occurring without voltage issues 1b - A previous read may have been corrupted due to read voltage being out of range
24 RRE	Read Reference Error RRE provides information on previous reads monitoring the read reference. If the read reference is detected to be out of range, this bit is set to indicate that previous reads requested may have been corrupted. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads occur without reference issues 1b - Previous read may be corrupted because of read reference being out of range
23-21 —	Reserved
20 RWE	Read-While-Write Event Error This bit provides information on previous read-while-write (RWW) reads. If an RWW error occurs, this bit is set to 1. The bit must then be cleared, or a reset must occur before it returns to a 0 state. If RWE is not set, or remains 0, this indicates that all previous RWW reads (from the last reset, or clearing of RWE) are correct. Since this bit is an error flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads occur normally 1b - RWW error occurred during a previous read
19-18 —	Reserved
17	Program and Erase Protection Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
PEP	PEP provides information about program and erase operations with respect to protection errors. A protection error occurs if a program is attempted to a locked sector or super sector, or if an erase is selected to a locked sector or super sector. This is evaluated prior to the operation beginning, and if an error is detected, high voltage operations (either a Program or Erase) will not be attempted for this request. **NOTE** If a location has both OTP and Lock protection, the response from the NVM will be PEP=1 only. If PEP is asserted, it must be cleared prior to attempting another high voltage operation. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Program and erase protection errors do not exist 1b - Previous program or erase protection error encountered
16 PES	Program and Erase Sequence Error PES provides information about program and erase operations with respect to sequence errors. A sequence error occurs when the program or erase sequence is not followed exactly. If an “out-of-order” event is detected, PES will assert, and the remainder of the sequence will not be accepted. PES monitoring begins when the PEADR register is written and ends when EHV is set to start the operation. PES monitoring only applies to Program and Erase (Sector and Block). It does not apply to Express Program, or UTest operations. Following is a complete list of sequence error conditions: • Any specific DATA register written twice. • More than one DATA register write for ERS operations. • Attempts to write PGM or ERS register out of sequence. • Attempts to write PGM after ERS, or ERS after PGM. • Attempts to write DATA registers after PGM or ERS are written. • Attempts to write EHV out of sequence. Writes to the MCR that are protected by the MCR Priority Levels (i.e. simultaneous PGM, ERS and EHV writes) will not result in a PES condition. Attempts to write MCR or DATA registers by a master that does not match the PEID, will not result in a PES condition. Those reads and writes will be blocked, and the master in control of the interface may continue with its' sequence. Clearing of PES (due to a sequence error) has the same effect as the clearing of PGM or ERS after a program or erase operation. PEADR and DATA registers are cleared, PEID is cleared, and the operation must be re-started from the beginning. If PGM or ERS are already set, they must be cleared before PES is cleared. PES can only be cleared once the PGM or ERS bit for the sequence in progress is cleared. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Program and erase sequence errors do not exist 1b - Previous program or erase sequence encountered an error
15 DONE	State Machine Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DONE indicates whether the embedded flash memory is performing a high voltage operation. DONE is set to a 1 on termination of the embedded flash memory reset. DONE is read only. DONE is set to a 1 at the end of program and erase high voltage sequences. **NOTE** This field transitions from a 0 to 1 during reset and remains at 1 after reset. 0b - Performing a high voltage operation 1b - Not executing a high voltage operation
14 PEG	Program/Erase Good The value of PEG is updated automatically during the program and erase high voltage operations. Aborting a program/erase high voltage operation causes PEG to be cleared, indicating the sequence failed. PEG is set to a 1 when the embedded flash memory is reset. PEG is read only. The value of PEG is valid only when PGM = 1 or ERS = 1 and after DONE transitions from 0 to 1 due to an abort or the completion of a program/erase operation. PEG is valid until PGM/ERS makes a 1 to 0 transition or EHV makes a 0 to 1 transition. **NOTE** 1. If program or erase operations are attempted on sector(s) that are locked, the response from embedded flash memory is PEG = 1, indicating that the operation was successful, and the contents of the sector(s) are properly protected from the program or erase operation. PEG = 1 is also true if an abort occurs during an HV request to a locked sector. 2. If a program or erase operation is attempted to a location marked as OTP, the response from the embedded flash memory is PEG = 0, indicating that the operation was not allowed. The value interlocked is not programmed, since desired double word was already programmed with a previous program operation. Erase is prevented on OTP locations. 0b - Program or erase operation failed 1b - Program or erase operation successful
13 —	Reserved
12 WDI	Watch Dog Interrupt WDI is a status register to indicate that the Watchdog Timer for Program or Erase had expired. WDI is status only, and will be automatically cleared once the operation that caused the watchdog timeout is terminated or clean up from the operation is completed (clearing of PGM (with PES low), ERS (with PES low) or PES bit). 0b - Normal Operation, Watchdog Timer has not expired. 1b - Program Watchdog Timer has expired.
11-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

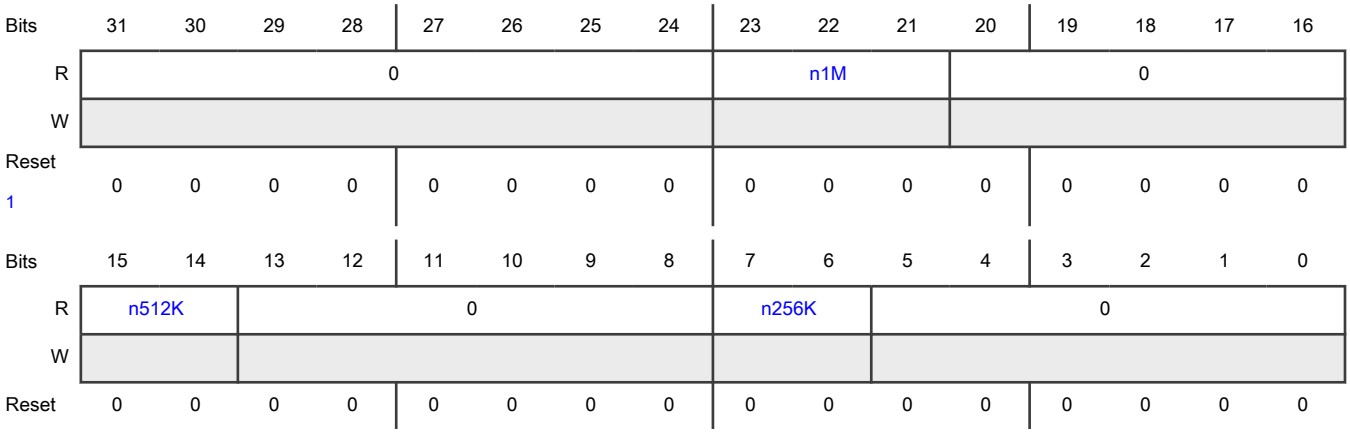
Field	Function
9 EPEG	ECC Enabled Program/Erase Good EPEG is a qualifier to PEG to indicate if a passing program or erase required ECC Enabled verifies to pass. In the event of a failed operation (PEG=0), EPEG will always remain 0. The value of EPEG is updated automatically during the program and erase high voltage operations. EPEG is set to a 0 when the embedded flash memory is reset. EPEG is read only. The value of EPEG is valid only when PGM = 1 or ERS = 1 and after DONE transitions from 0 to 1 due to an abort or the completion of a program/erase operation. EPEG is valid until PGM/ERS makes a 1 to 0 transition or EHV makes a 0 to 1 transition. 0b - Program or erase operation did not require ECC Enabled verifies. 1b - Program or erase operation required ECC Enabled verifies to pass.
8 TSPELOCK	UTest NVM Program and Erase Lock TSPELOCK reflects the status of the hardware program and erase protection input to the flash module. The TSPELOCK register is latched on the PEADR[PEASAD] write (UTest Sector Interlock), is not writable, and is status only. During high voltage events, the status is locked. When not interlocked this bit will read 1. The default value of the TSPELOCK bits is program and erase protected. 0b - Corresponding sector not locked, and may be programmed or erased 1b - Corresponding sector protected from the program and erase sequences
7-1 —	Reserved
0 RE	Reset Error This bit provides information on previous resets. Checks are done within the flash and if a coherency issue or ECC error is detected during the reset reads used for trimming on the flash, this status flag will assert. This bit is status only, and is not writable. 0b - Reset occurred without errors 1b - Reset error encountered

20.7.1.4 Extended Module Configuration (MCRE)

Offset

Register	Offset
MCRE	8h

Diagram



1. The reset value of this register is set at the factory and varies among chips. See the chip-specific c40asf information for details.

Fields

Field	Function
31-24 —	Reserved
23-21 n1M	Number of 1 MB Blocks 000b - Zero 1 MB blocks 001b - One 1 MB block 010b - Two 1 MB blocks 011b - Three 1 MB blocks 100b - Four 1 MB blocks 101b - Reserved 110b - Reserved 111b - Reserved
20-16 —	Reserved
15-14 n512K	Number of 512 KB Blocks 00b - Zero 512 KB blocks 01b - One 512 KB block 10b - Two 512 KB blocks 11b - Four 512 KB blocks
13-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

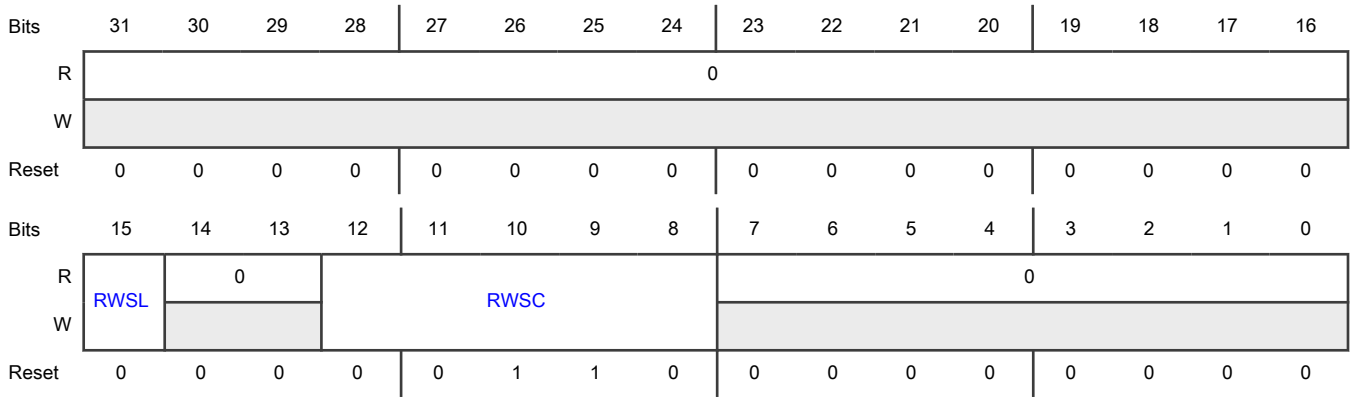
Field	Function
7-6 n256K	Number of 256 KB Blocks 00b - Zero 256 KB blocks 01b - One 256 KB block 10b - Two 256 KB blocks 11b - Four 256 KB blocks
5-0 —	Reserved

20.7.1.5 Module Control (CTL)

Offset

Register	Offset
CTL	Ch

Diagram



Fields

Field	Function
31-16 —	Reserved
15 RWSL	Read Wait State Lock Read Wait State Lock. The RWSL bit locks the read wait state control register (RWSC). If RWSL is set, then the RWSC field can not be written. RWSL is not clearable by a register write, it is only cleared upon reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Once RWSL is written to a 1, it will remain a 1 until the next reset and RWSC will be locked from writing. RWSC and RWSL are allowed to be written simultaneously. RWSL does not affect readability of RWSC. 0b - RWSC not locked and available for writing 1b - RWSC locked and unavailable for writing
14-13 —	Reserved
12-8 RWSC	Wait State Control This field controls the number of wait states to be added to the best-case flash array access time for reads. RWSC is only writable when RWSL is a 0. The best case flash array access time for reads is one cycle. This field must be set to a value corresponding to the operation frequency of the Flash and the actual read access time of the Flash to the flash boundary. Total read latency will be SoC dependant based on the how the cores are coupled within the platform. The frequency information, and total latency details are documented in the SoC specification. Higher operating frequencies require non-zero settings for this field for proper flash operation. **NOTE** • Updates to this configuration field that control reads must take place only when the flash memory is idle. Changing configuration settings while flash memory access is in progress can lead to non-deterministic behavior. • Values of RWSC[4:0] not listed here are reserved. **NOTE** 0_0000b - Reserved 0_0001b - One additional wait state is added. 0_0010b - Two additional wait states are added. 0_0011b - Three additional wait states are added. 0_0100b - Four additional wait states are added. 0_0101b - Five additional wait states are added. 0_0110b - Six additional wait states are added. 0_0111b - Seven additional wait states are added. 0_1000b - Eight additional wait states are added. 0_1001b - Reserved 1_1111b - Reserved
7-0 —	Reserved

20.7.1.6 Address (ADR)

Offset

Register	Offset
ADR	10h

Function

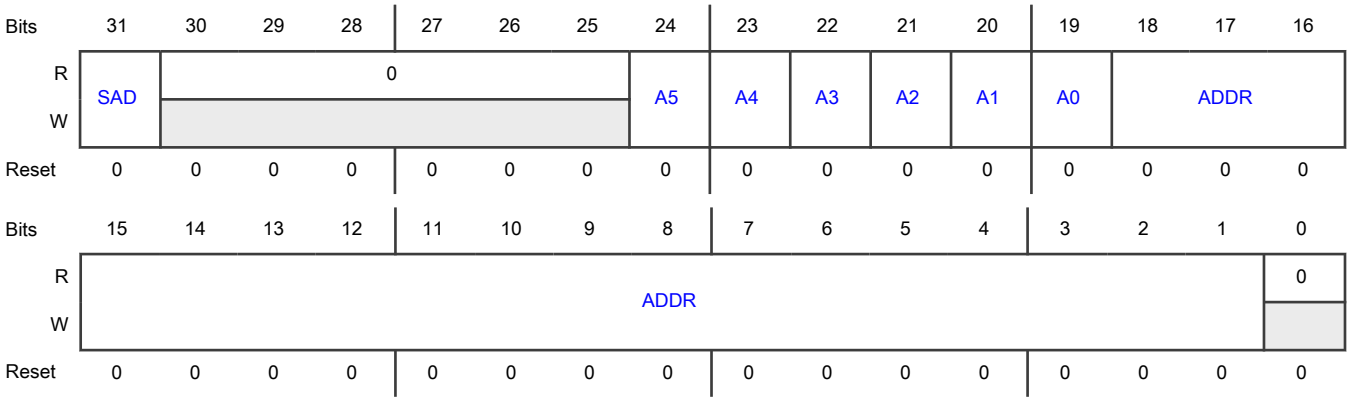
The Address register (ADR) provides the first failing address in the event of a failure (ECC or PGM/Erase state machine), as well as single bit correction information. The address register is also writable for UTest Mode operations. The address is identified using a combination of bits that identify the memory region and offset address.

NOTE

- The address given is an offset from the base address of the address space.
- The block numbering scheme that corresponds to the offset address is based on the flash memory's *internal* addressing scheme, which may be different than the chip's system addressing scheme.

Understanding the mapping between the flash address, which is specified relative to the flash module's internal addressing scheme, and the location within the chip's system memory map requires a clear understanding of the flash memory layout. See the chip-specific information for a detailed explanation of the chip's flash memory layout and how to map system addresses to the flash module's internal addressing.

Diagram



Fields

Field	Function
31 SAD	UTest NVM Address Qualifies the address captured during an ECC event error, single bit correction, or state machine operation. See the description of the ADDR field for more information. This field is not writeable if UTE = 1, and is ignored if it's in UTest mode. 0b - Address captured or to be accessed is from the main array space 1b - Address captured or to be accessed is from the UTest NVM array space

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-25 —	Reserved Reserved; reset to 0
24 A5	Address Region 5 Qualifies the address field (ADDR) to region 5 If the value of this field is 1, the ADDR field maps to region 5. See the description of the ADDR field for more information. If the region is not present, this field is locked to 0. 0b - Address captured or to be accessed is not from region 5 1b - Address captured or to be accessed is from region 5
23 A4	Address Region 4 Qualifies the address field (ADDR) to region 4 If the value of this field is 1, the ADDR field maps to region 4. See the description of the ADDR field for more information. If the region is not present, this field is locked to 0. 0b - Address captured or to be accessed is not from region 4 1b - Address captured or to be accessed is from region 4
22 A3	Address Region 3 Qualifies the address field (ADDR) to region 3 If the value of this field is 1, the ADDR field maps to region 3. See the description for the ADDR field for more information. If the region is not present, this field is locked to 0. 0b - Address captured or to be accessed is not from region 3 1b - Address captured or to be accessed is from region 3
21 A2	Address Region 2 Qualifies the address field (ADDR) to region 2 If the value of this field is 1, the ADDR field maps to region 2. See the description of the ADDR field for more information. If the region is not present, this field is locked to 0. 0b - Address captured or to be accessed is not from region 2 1b - Address captured or to be accessed is from region 2
20 A1	Address Region 1 Qualifies the address field (ADDR) to the region If the value of this field is 1, the ADDR field maps to region 1. See the description for the ADDR field for more information. If the region is not present, this field is locked to 0. 0b - Address captured or to be accessed is not from region 1 1b - Address captured or to be accessed is from region 1
19	Address Region 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
A0	Qualifies the ADDR field to region 0. If A0 = 1, then the ADDR field maps to that region. If the region is not present, this field is locked to 0. For details, see the field description for ADDR. 0b - Address captured or to be accessed is not from region 0 1b - Address captured or to be accessed is from region 0
18-1 ADDR	Address The ADR register provides the first failing address in the event of ECC event error (EER set), single bit correction (SBC set), as well as providing the address of a failure that may have occurred in a state machine operation (PEG cleared). ECC event errors take priority over single bit corrections, which take priority over state machine errors. This is especially valuable in the event of an RWW operation, where the read senses an ECC error or single bit correction, and the state machine fails simultaneously. The failing address for ECC event error is held until EER is cleared. The failing address for single bit correction is held until an ECC event error, or until SBC is cleared. State machine address is held until an ECC event error or single bit correction event occurs, or until the next state machine event (PEG cleared) occurs. This address is always a double word address that selects 64 bits. The ADR register is writable, and can be used in the UTest ECC Logic Test, UTest EDC after ECC Logic Test, and UTest Address Encode Logic Test. If any of these tests are enabled (UT0[EIE] = 1, UT0[EDIE] = 1, or UT0[AEIE] = 1) then the ADR register will not update for ECC event error, single bit correction or state machine errors. If MCRS[EER] or MCRS[SBC] are set, the ADR register is locked from writing. MCRS[PEG] does not affect the writeability of the ADR register. ADDR[18:1] is an offset from a base address of 0h for each block region. The A0, A1, A2, A3, A4, and A5 bits qualify the block size region the ADDR field.
0 —	Reserved Reserved; reset to 0

20.7.1.7 Program and Erase Address (PEADR)

Offset

Register	Offset
PEADR	14h

Function

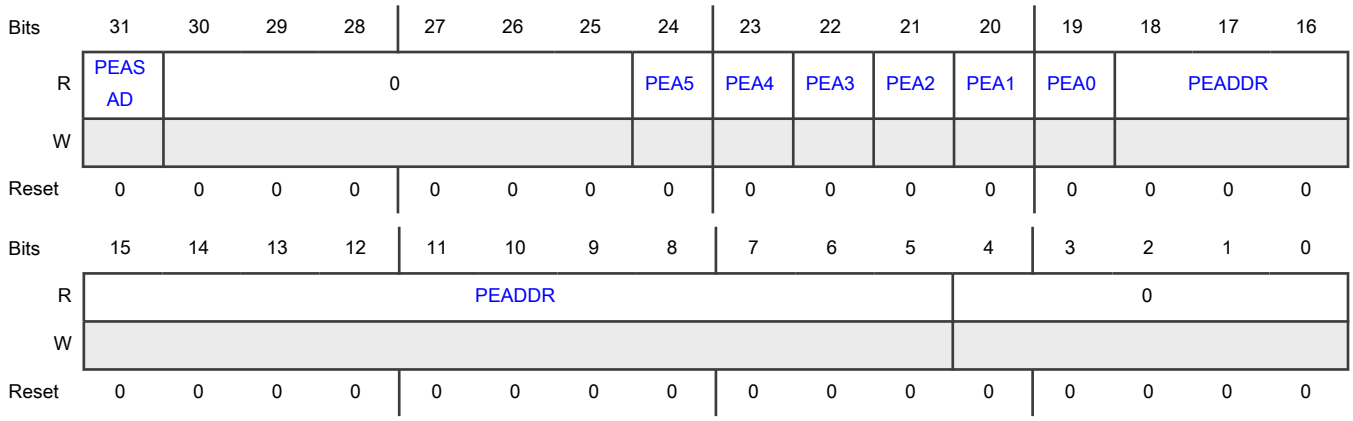
The Program and Erase Address register (PEADR) is used to provide the address to be programmed or the location of the sector or block to be erased. The program and erase address register is read only in user mode (user mode writes occur with writes done to the Platform Flash Controller). The address is identified using a combination of bits that identify the memory region and offset address.

NOTE

- The address provided is an offset from the base address of the address space.
- The block numbering scheme that corresponds to the offset address is based on the flash memory's *internal* addressing scheme, which may be different from the chip's system addressing scheme.

Understanding the mapping between the flash address, which is specified relative to the flash module's internal addressing scheme, and the location within the chip's system memory map requires a clear understanding of the flash memory layout. See the chip-specific information for a detailed explanation of the chip's flash memory layout and how to map system addresses to the flash module's internal addressing.

Diagram



Fields

Field	Function
31 PEASAD	UTest NVM Program and Erase Address Qualifies the address field (PEADDR) to the region. If PESAD = 1, the PEADDR field maps to that region. See the description of the PEADDR field for more information. This field is not writeable if UTE = 1 and is ignored if it becomes 1 while in UTest mode. 0b - Address accessed is from the main array space 1b - Address accessed is from the UTest NVM array space
30-25 —	Reserved Reserved; reset to 0
24 PEA5	Program and Erase Address Region 5 Qualifies the address field (PEADDR) to region 5. If PEA5 = 1, the PEADDR field maps to region 5. See the description of the PEADDR field for more information. If region 5 is not present, this field is locked to 0. 0b - Address accessed is not from region 5 1b - Address accessed is from region 5

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 PEA4	Program and Erase Address Region 4 Qualifies the address field (PEADDR) to region 4. If PEA4 = 1, the PEADDR field maps to region 4. See the description of the PEADDR field for more information. If region 4 is not present, this field is locked to 0. 0b - Address accessed is not from region 4 1b - Adrress accessed is from region 4
22 PEA3	Program and Erase Address Region 3 Qualifies the address field (PEADDR) to region 3. If PEA3 = 1, the PEADDR field maps to region 3. See the description for PEADDR for more information. If region 3 is not present, this field is locked to 0. 0b - Address accessed is not from region 3 1b - Address accessed is from region 3
21 PEA2	Program and Erase Address Region 2 Qualifies the address field (PEADDR) to the region. If PEA2 = 1, the PEADDR field maps to region 2. See the description of the PEADDR field for more information. If region 2 is not present, this field is locked to 0. 0b - Address accessed is not from region 2 1b - Address accessed is from region 2
20 PEA1	Program and Erase Address Region 1 Qualifies the address field (PEADDR) to the region. If PEA1 = 1, the PEADDR field maps to region 1. See the description of the PEADDR field for more information. If region 1 is not present, this field is locked to 0. 0b - Address accessed is not from region 1 1b - Address accessed is from region 1
19 PEA0	Program and Erase Address Region 0 Qualifies the address field (PEADDR) to the region. If PEA0 = 1, the PEADDR field maps to region 0. See the description of the PEADDR field for more information. If region 0 is not present, this field is locked to 0. 0b - Address accessed is not from region 0 1b - Address accessed is from region 0
18-5 PEADDR	Program and Erase Address The PEADDR register provides offset address location to be programmed or the sector to be erased in the case of sector erase. This address is always a quad page address that selects 1024 bits.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The PEADDR register is read only in user mode and represents the flash physical address status. PEADDR may be updated once (and only once) per Program or Erase event. PEADDR writes initiate a Program, Erase, Array Integrity or User Margin Read request. Once PEADDR is updated, it will be locked for the full program and erase operation, until MCR[PGM] or MCR[ERS] is cleared at the end of the operation (as long as MCRS[PES] is zero). If MCRS[PES] is set, PEADR is locked until this bit is also cleared. **NOTE** If Express Program has been initiated (XPEADR written), the PEADR is locked from writing until Express Program is completed (XPGM written to 0). Upon completion of an operation (Program (MCR[PGM] written to 0 and MCRS[PES] equals 0), Erase (MCR[ERS] written to 0 and MCRS[PES] equals 0), Array Integrity (UT0[AIE] written to 0), User Margin Read(UT0[AIE] written to 0)), or Program and Erase Sequence clear (MCRS[PES] written to 0) the PEADR registers will return to an all zeros state. When MCR[EHV] or UT0[AIE] are set, PEADDR bits will not be writeable. Once PEADDR is updated, it will have read restrictions based on the PEID master that did the original write in the Platform Flash Controller. PEADDR[18:5] is an offset from a base address of 0h for each block region. The PEA0, PEA1, PEA2, PEA3, PEA4 and PEA5 qualify the block size region the PEADDR field.
4-0 —	Reserved Reserved; reset to 0

20.7.1.8 Sector Program and Erase Hardware Lock (SPELOCK)

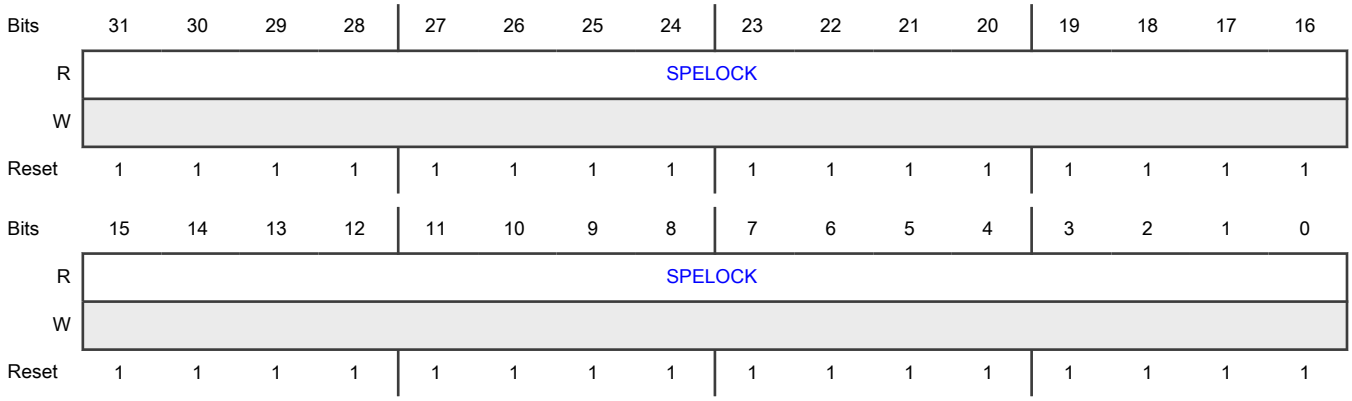
Offset

Register	Offset
SPELOCK	50h

Function

The Sector Program and Erase Lock register provides a means to protect sectors from being programed and erased on the main interface. The last 256 KB of all blocks have sector lock protection capability. The rest of the block (if applicable) is protected with super sector protection capability. This register shows the status of the pelock hardware input for the block that is interlocked. For more information on hardware program and erase protection see Program hardware locking and Erase hardware locking .

Diagram



Fields

Field	Function
31-0 SPELOCK	Sector Program and Erase Lock [31:0] SPELOCK reflects the status of the hardware program and erase protection input to the flash module for the block interlocked on the main interface. The SPELOCK register is latched on the PEADR write, is not writable, and is status only. SPELOCK will return back to all ones at the same time that PEADR is cleared. When not interlocked this register will read all ones. During high voltage events, the status is locked. The default value of the SPELOCK bits is program and erase protected. In the event that sectors are not present (due to configuration or total memory size), the SPELOCK bits default to locked, and will remain 1. 0 - The corresponding sector is not locked, and may be programed or erased. 1 - The corresponding sector is protected from program or erase.

20.7.1.9 Super Sector Program and Erase Hardware Lock (SSPELOCK)

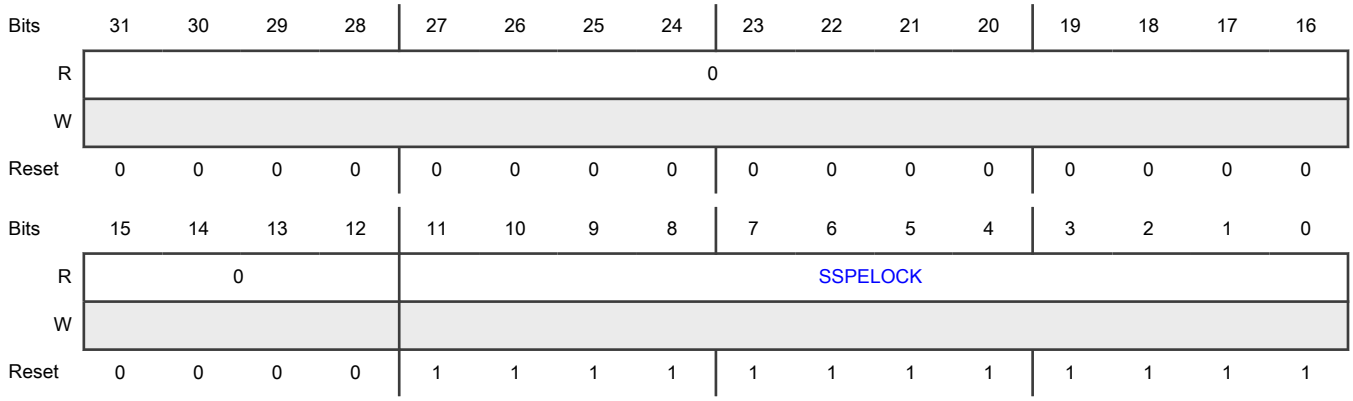
Offset

Register	Offset
SSPELOCK	54h

Function

The Program and Erase Lock register provides a means to protect super sectors from being programed and erased on the main interface. Super Sector protection is available on block space greater than 256 KB. For 256 KB blocks, this register is not applicable. For 512 KB blocks, the first half of the block is protected with super sector granularity. For 1 MB blocks, the first 768 KB is protected with super sector granularity. For 2 MB blocks, the first 1792 KB is protected with super sector granularity. This register shows the status of the pelock hardware input for the block interlocked. For more information on hardware program and erase protection see Program hardware locking and Erase hardware locking .

Diagram



Fields

Field	Function
31-12 —	Reserved Reserved; reset to 0
11-0 SSPELOCK	Super Sector Program and Erase Lock [11:0] SSPELOCK reflects the status of the hardware program and erase protection input to the flash module for the block interlocked on the main interface. The SSPELOCK register is latched on the PEADR write, is not writable, and is status only. SSPELOCK will return back to all ones at the same time that PEADR is cleared. When not interlocked this register will read all ones. During high voltage events, the status is locked. The default value of the SSPELOCK bits is program and erase protected. In the event that super sectors are not present (due to configuration or total memory size), or super sectors not present due to block size, the SSPELOCK bits default locked, and will remain 1. 0 - The corresponding super sector is not locked, and may be programmed or erased. 1 - The corresponding super sector is protected from program or erase.

20.7.1.10 Express Sector Program and Erase Hardware Lock (XSPELOCK)

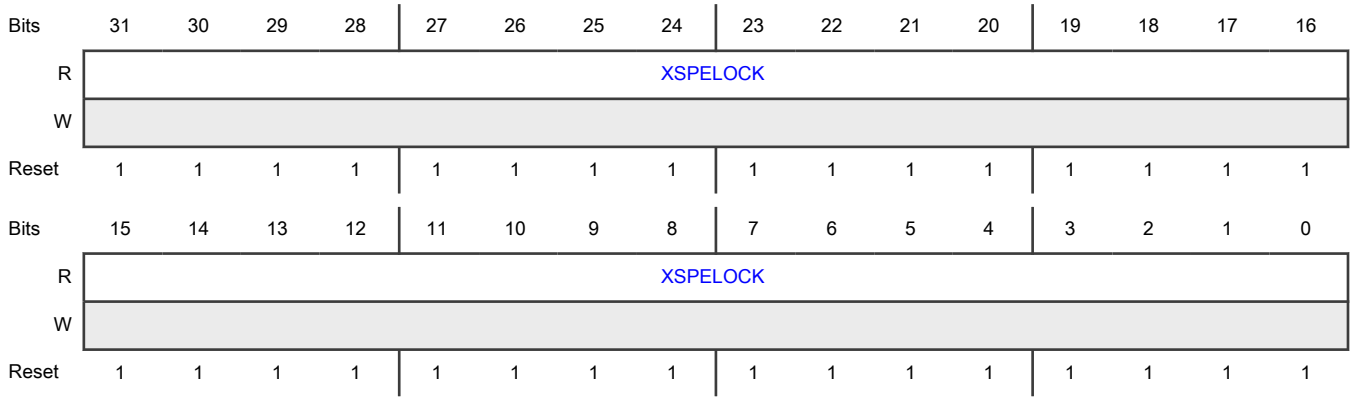
Offset

Register	Offset
XSPELOCK	70h

Function

The Express Program and Erase Lock register provides a means to protect sectors from being programed and erased on the express program interface. The last 256 KB of all blocks have sector lock protection capability. The rest of the block (if applicable) is protected with super sector protection capability. This register shows the status of the pelock hardware input for the block that is interlocked. For more information on hardware program and erase protection see Program hardware locking and Erase hardware locking .

Diagram



Fields

Field	Function
31-0 XSPELOCK	Express Sector Program and Erase Lock [31:0] XSPELOCK reflects the status of the hardware program and erase protection input to the flash module for the block interlocked on the express program interface. The XSPELOCK register is latched on the XPEADR write, is not writable, and is status only. XSPELOCK will return back to all ones at the same time that XPEADR is cleared. When not interlocked this register will read all ones. During high voltage events, the status is locked. The default value of the XSPELOCK bits is program and erase protected. In the event that sectors are not present (due to configuration or total memory size), the XSPELOCK bits default to locked, and will remain 1. 0 - The corresponding sector is not locked, and may be programmed or erased. 1 - The corresponding sector is protected from program or erase.

20.7.1.11 Express Super Sector Program and Erase Hardware Lock (XSSPELOCK)

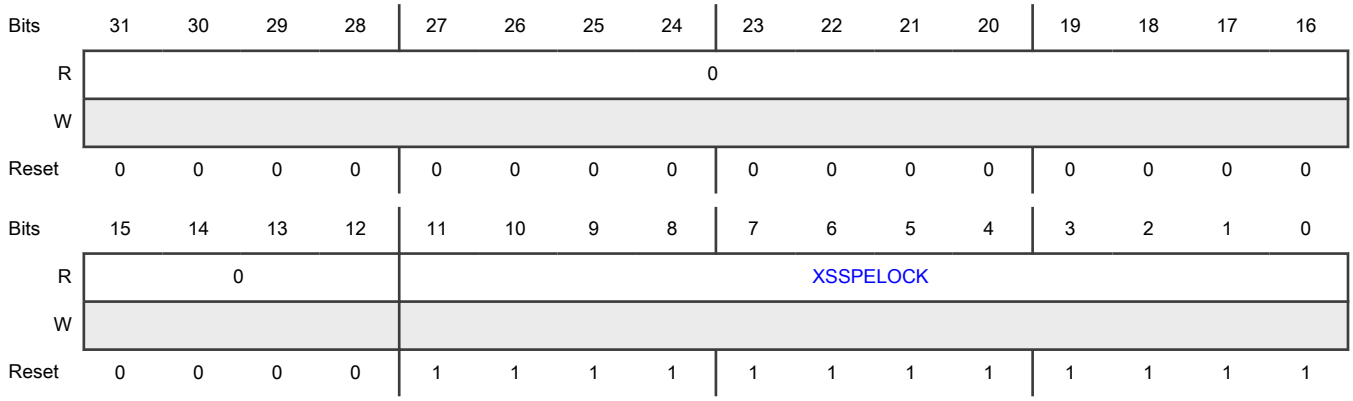
Offset

Register	Offset
XSSPELOCK	74h

Function

The Express Program and Erase Lock register provides a means to protect super sectors from being programed and erased on the express program interface. Super Sector protection is available on block space greater than 256 KB. For 256 KB blocks, this register is not applicable. For 512 KB blocks, the first half of the block is protected with super sector granularity. For 1 MB blocks, the first 768 KB is protected with super sector granularity. For 2 MB blocks, the first 1792 KB is protected with super sector granularity. This register shows the status of the pelock hardware input for the block interlocked. For more information on hardware program and erase protection see Program hardware locking and Erase hardware locking .

Diagram



Fields

Field	Function
31-12 —	Reserved Reserved; reset to 0
11-0 XSSPELOCK	Express Super Sector Program and Erase Lock [11:0] XSSPELOCK reflects the status of the hardware program and erase protection input to the flash module for the block interlocked on the express program interface. The XSSPELOCK register is latched on the XPEADR write, is not writable, and is status only. XSSPELOCK will return back to all ones at the same time that XPEADR is cleared. When not interlocked this register will read all ones. During high voltage events, the status is locked. The default value of the XSSPELOCK bits is program and erase protected. In the event that super sectors are not present (due to configuration or total memory size), or super sectors not present due to block size, the XSSPELOCK bits default to locked, and will remain 1. 0 - The corresponding super sector is not locked, and may be programmed or erased. 1 - The corresponding super sector is protected from program or erase.

20.7.1.12 Test Mode Disable Password Check (TMD)

Offset

Register	Offset
TMD	90h

Function

Provides a means to supply a challenge password to disable the Test mode disable seal block select.

Writes to the register have no effect except for a password challenge. Reads to this register always return 0.

Diagram

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

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

Fields

Field	Function
31-0 PWD	Password Challenge

20.7.1.13 UTest 0 (UT0)

Offset

Register	Offset
UT0	94h

Function

Provides a means to control UTest.

UTest mode provides the ability to perform test features on the flash memory. This register is only writeable when the flash memory is put into UTest mode by writing a passcode.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RRIE	AEIE	EDIE	EIE	0		NAIBP	AIBPE	0		AISUS	MRE	MRV	0		AID
W															AIS	AIE	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31 UTE	UTest Enable This status bit indicates when U-Test is enabled. All bits in UT0, UM0, UM1, UM2, UM3, UM4, UM5, UM6, UM7, UM8, UM9, UD0, UD1, UD2, UD3, UD4, UD5, UA0 and UA1 registers are locked when this bit is 0. This bit is not writeable to a 1, but may be cleared. The reset value is 0. The method to set this bit is to provide a password, and if the password matches, the UTE bit is set to reflect a status of enabled, and is enabled until it is cleared by a register write. The UTE password (set UTE=1) and UTE clearing to 0 will only be accepted if Program or Erase are not in progress (PEADR write through clearing of PGM/ERS/PES), Alternate Program or Erase are not in progress (APEADR write through clearing of APGM/AERS/APES), AI/MR Operation are not in progress (PEADR write through clearing of AIE), and an Express Program operation is not in progress (XPEADR write through the clearing of XPGM). UTE can only be cleared if AID = 1, MRE, AIE, EIE, EDIE, AEIE, and RRIE are all equal to 0. While successfully clearing UTE, writes to set AIE, MRE, EIE, EDIE, AEIE or RRIE will be ignored. For UTE, the password F9F9_9999h must be written to the UT0 register, and this must be a 32bit write. 0b - U-Test mode is not enabled. 1b - U-Test mode is enabled.
30 SBCE	Single Bit Correction Enable SBCE enables single bit correction results to be observed in SBC. ECC corrections that occur when SBCE is cleared will not be logged. 0b - Disabled 1b - Enabled
29-16 —	Reserved Reserved; reset to 0
15 RRIE	Read Reference Input Enable RRIE enables the force of an error on the read reference detection circuit. This is useful in the Read Reference Error check. If this bit is set, the read reference error register MCRS[RRE] will set when an address match is achieved to the ADR register. RRIE is not simultaneously writable to a 1 as UTE is being successfully cleared to a 0. 0b - Read reference input disabled 1b - Read reference input enabled
14 AEIE	Address Encode Invert Enable AEIE enables the input register AEI to invert the address encode value received from the array, and force a mismatch in the address encode comparison. This is useful in the Address Encode logic check. If this bit is set, address encode information from the memory array will be inverted based on values in the AEI register when an address match is achieved to the ADR register. AEIE is not simultaneously writable to a 1 as UTE is being successfully cleared to a 0. 0b - Address encode invert is disabled 1b - Address encode values are inverted based on UA0[AEI]
13	EDC after ECC Data Input Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
EDIE	EDIE enables the input registers EDDATA and EDDATAC to be the source of data to the EDC after ECC comparator or EDC after ECC encoder. This is useful in the EDC after ECC logic check. If this bit is set, data read via a BIU read request will be from the EDDATA and EDDATAC registers when an address match is achieved to the ADR register. EDIE is not simultaneously writable to a 1 as UTE is being successfully cleared to a 0. 0b - EDC after ECC data input is disabled 1b - Data read is from UD3[EDDATA] and UD5[EDDATAC]
12 EIE	ECC Data Input Enable EIE enables the input registers EDATA and EDATAC to be the source of data to the ECC logic (instead of from the memory array). This is useful in the ECC logic check. If this bit is set, data read via a BIU read request will be from the EDATA and EDATAC registers when an address match is achieved to the ADR register. EIE is not simultaneously writable to a 1 as UTE is being successfully cleared to a 0. 0b - ECC data input is disabled 1b - Data read is from UD0[EDATA] and UD2[EDATAC]
11-10 —	Reserved
9 NAIBP	Next Array Integrity Break Point If AIBPE is set, NAIBP will be set once a single bit correction (if enabled) or double bit detection is noted during the Array Integrity test. NAIBP is not writable to 1, but may be cleared to 0. Clearing NAIBP to 0, will enable the Array Integrity operation to be re-started after a breakpoint is encountered. NAIPB may only be cleared to 0 if both EER = 0 and SBC = 0. If the Array Integrity operation completes without encountering another correction or detection, AID will be set with NAIBP remaining 0. 0b - Array integrity state machine is not currently at a breakpoint 1b - Array integrity state machine is at a breakpoint
8 AIBPE	Array Integrity Break Point Enable To enable breakpoints during an array integrity test, AIBPE may be set. See NAIBP description for more information about array integrity breakpoints. 0b - Array integrity breakpoints disabled 1b - Array integrity breakpoints enabled during array integrity checks
7 —	Reserved Reserved; reset to 0
6 AISUS	Array Integrity Suspend AISUS enables a suspend of an Array Integrity Operation. Array Integrity may be suspend by setting AISUS, and resumed by clearing AISUS. AISUS is writeable to a 1 only when AID is low. AISUS is clearable to a 0 only when AID is high. Attempting to write AISUS and AIE on the same clock cycle will result in only AIE getting written.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Array integrity sequence not suspended. 1b - Array integrity sequence is suspended.
5 MRE	Margin Read Enable MRE combined with MRV enables user margin reads to be done. Normal user reads are not affected by MRE, although user reads while the margin read operation is ongoing are not supported. MRE is not writable if AID is low, or if AISUS is high, or if NAIBP is high. MRE is not simultaneously writable to a 1 as UTE is being successfully cleared to a 0. 0b - Margin reads are not enabled. 1b - Margin reads are enabled.
4 MRV	Margin Read Value MRV selects the margin level that is being checked. Margin can be checked to an erased level (MRV=1) or to a programmed level (MRV=0). In order for this value to be valid, MRE must also be set. MRV is not writable if AID is low, or if AISUS is high, or if NAIBP is high. 0b - Zero's margin reads are requested. 1b - One's margin reads are requested.
3 —	Reserved
2 AIS	Array Integrity Sequence AIS determines the address sequence to be used during array integrity checks. The default sequence (AIS = 0) is meant to replicate sequences that normal user code follows, and thoroughly checks the read propagation paths. This sequence is proprietary. The alternative sequence (AIS = 1) is logically sequential. It should be noted that the time to run a sequential sequence is shorter than the time to run the proprietary sequence. AIS is not writeable if AIE is high. 0b - Array integrity sequence is proprietary sequence 1b - Array integrity sequence is sequential
1 AIE	Array Integrity Enable AIE set to one starts the array integrity check done on all selected blocks. The address sequence is determined by AIS, and the MISR (UM0 through UM9) can be checked after the operation is complete, to determine if a correct signature has been obtained. Once an array integrity operation is requested (AIE=1), it may be terminated by clearing AIE if the operation has finished (AID = 1) or aborted by clearing AIE if the operation is ongoing (AID = 0). AIE is locked from writing unless an interlock write occurred (PEADR write and DATA write). AIE is not simultaneously writable to a 1 as UTE is being successfully cleared to a 0. 0b - Array integrity checks not enabled 1b - Array integrity checks enabled
0	Array Integrity Done

Table continues on the next page...

Table continued from the previous page...

Field	Function
AID	AID is cleared upon an array integrity check being enabled (to signify the operation is ongoing). Once completed, AID is set to indicate that the array integrity check is complete. At this time the MISR (UMR registers) can be checked. AID may also assert if breakpoints are enabled (AIBPE is set), an abort is requested or a suspend is requested. AID cannot be written, and is status only. 0b - Array integrity check ongoing 1b - Array integrity check complete

20.7.1.14 UMISRn (UM0 - UM8)

Offset

For a = 0 to 8:

Register	Offset
UMa	98h + (a × 4h)

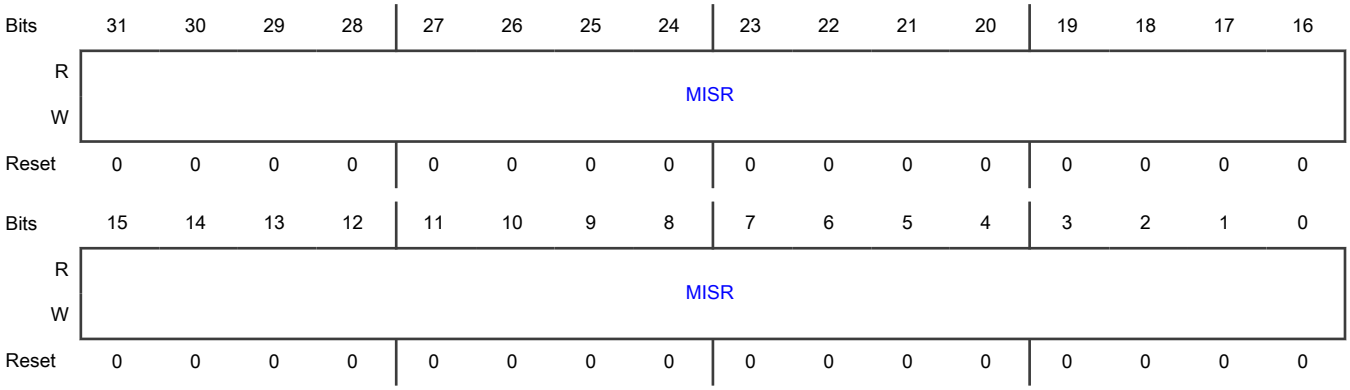
Function

Provides a means to evaluate array integrity. UM9 to UM0 are a set of multiple-input signature registers (MISRs) that hold the 289-bit MISR value as shown in the next table.

Table 89. MISR mapping

UM register	MISR bits
UM0	MISR[31:0]
UM1	MISR[63:32]
UM2	MISR[95:64]
UM3	MISR[127:96]
UM4	MISR[159:128]
UM5	MISR[191:160]
UM6	MISR[223:192]
UM7	MISR[255:224]
UM8	MISR[287:256]
UM9	MISR[288]

Diagram



Fields

Field	Function
31-0 MISR	MISR[31:0] The MISR registers accumulate a signature from an array integrity event. The MISR captures all data fields, as well as ECC fields, and the ECC error signal. Data (read and ECC) captured in the MISR is after the ECC logic, and represents corrected data (if required). The MISR can be seeded to any value by writing the MISR registers. **NOTE** Writing the MISR register during an array integrity operation (including suspend or breakpoint) although possible, is not recommended. A write of the MISR registers at this point may alter the final signature in an unpredictable way. Writing the MISR registers prior to an Array Integrity operation (to seed the MISR) is allowed. The MISR register provides a means to calculate a MISR during array integrity operations. The MISR can be represented by the following polynomial: $x^{289} + x^{7} + x^{6} + x^{5} + x^{4} + x^{2} + 1$ The MISR is calculated by taking the previous MISR value and then "exclusive ORing" the new data. In addition the most significant bit (in this case it is MISR[288]), is then "exclusive ORed" into input of MISR[7], MISR[6], MISR[5], MISR[4], MISR[2] and MISR[0]. The result of the "exclusive OR" is shifted left on each read. The MISR register is used in array integrity operations. If during address sequencing, reads extend into an invalid address location (in other words, greater than the maximum address for a given array size) then reads are still executed to the array, but the results from the array read are not deterministic. In this instance, the MISR registers are not recalculated, and the previous value is retained. Once the AIE register is cleared, the MISR register will return to an all 0's state.

20.7.1.15 UMISR9 (UM9)

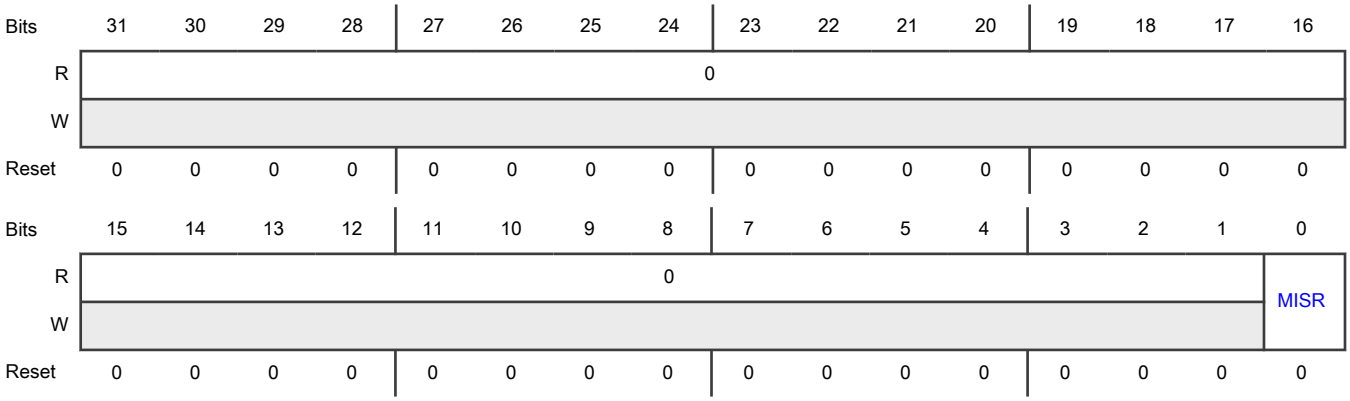
Offset

Register	Offset
UM9	BCh

Function

Provides a means to evaluate array integrity.

Diagram



Fields

Field	Function
31-1 —	Reserved Reserved; reset to 0
0 MISR	MISR[288] The MISR registers accumulate a signature from an array integrity event. The MISR captures all data fields, as well as ECC fields, and the ECC error signal. Data (read and ECC) captured in the MISR is after the ECC logic, and represents corrected data (if required). The MISR can be seeded to any value by writing the MISR registers. **NOTE** Writing the MISR register during an array integrity operation (including suspend or breakpoint) although possible, is not recommended. A write of the MISR registers at this point may alter the final signature in an unpredictable way. Writing the MISR registers prior to an Array Integrity operation (to seed the MISR) is allowed. The MISR register provides a means to calculate a MISR during array integrity operations. The MISR can be represented by the following polynomial: $x^{289} + x^7 + x^6 + x^5 + x^4 + x^2 + 1$

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The MISR is calculated by taking the previous MISR value and then "exclusive ORing" the new data. In addition the most significant bit (in this case it is MISR[288]), is then "exclusive ORed" into input of MISR[7], MISR[6], MISR[5], MISR[4], MISR[2] and MISR[0]. The result of the "exclusive OR" is shifted left on each read. The MISR register is used in array integrity operations. If during address sequencing, reads extend into an invalid address location (in other words, greater than the maximum address for a given array size) then reads are still executed to the array, but the results from the array read are not deterministic. In this instance, the MISR registers are not recalculated, and the previous value is retained. Once the AIE register is cleared, the MISR register will return to an all 0's state.

20.7.1.16 UTest Data 0 (UD0)

Offset

Register	Offset
UD0	D0h

Function

Provides a means to enable data to be substituted during UTest procedures.

Diagram

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

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

Fields

Field	Function
31-0 EDATA	ECC Data [31:0] Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. EDATA corresponds to the 32 array bits representing word 0 of all double words in the page selected in ADR.

20.7.1.17 UTest Data 1 (UD1)

Offset

Register	Offset
UD1	D4h

Function

Provides a means to enable data to be substituted during UTest procedures.

Diagram

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

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

Fields

Field	Function
31-0 EDATA	ECC Data [63:32] Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. UD0[EDATA] corresponds to the 32 array bits representing word 1 of all double words in the page selected in ADR.

20.7.1.18 UTest Data 2 (UD2)

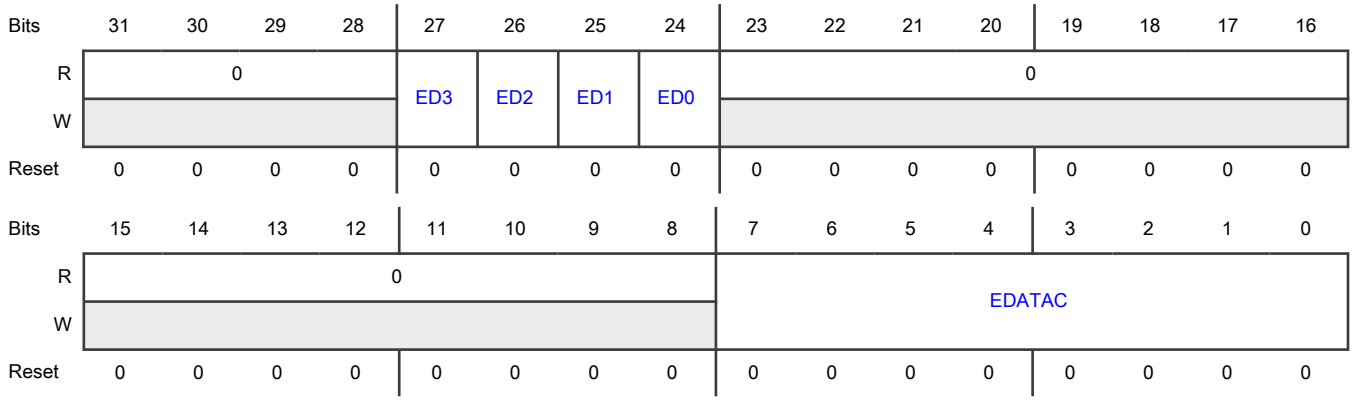
Offset

Register	Offset
UD2	D8h

Function

Provides a means to enable data to be substituted during UTest procedures.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 ED3	ECC Logic Check Double Word 3 Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW3 on the page selected in ADR.
26 ED2	ECC Logic Check Double Word 2 Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW2 on the page selected in ADR.
25 ED1	ECC Logic Check Double Word 1 Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW1 on the page selected in ADR.
24 ED0	ECC Logic Check Double Word 0 Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW0 on the page selected in ADR.
23-8 —	Reserved
7-0 EDATAC	ECC Data Check Bits [7:0] Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field corresponds to the eight array bits representing the check bits of all double words in the page selected in ADR.

20.7.1.19 UTest Data 3 (UD3)

Offset

Register	Offset
UD3	DCh

Function

Provides a means to enable data to be substituted during UTest procedures.

Diagram

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

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

Fields

Field	Function
31-0 EDDATA	EDC After ECC Data [31:0] Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. This field corresponds to the 32 array bits representing word 0 of all double words in the page selected in ADR.

20.7.1.20 UTest Data 4 (UD4)

Offset

Register	Offset
UD4	E0h

Function

Provides a means to enable data to be substituted during UTest procedures.

Diagram

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

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

Fields

Field	Function
31-0 EDDATA	EDC After ECC Data [63:31] Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. This field corresponds to the 32 array bits representing word 1 of all double words in the page selected in ADR.

20.7.1.21 UTest Data 5 (UD5)

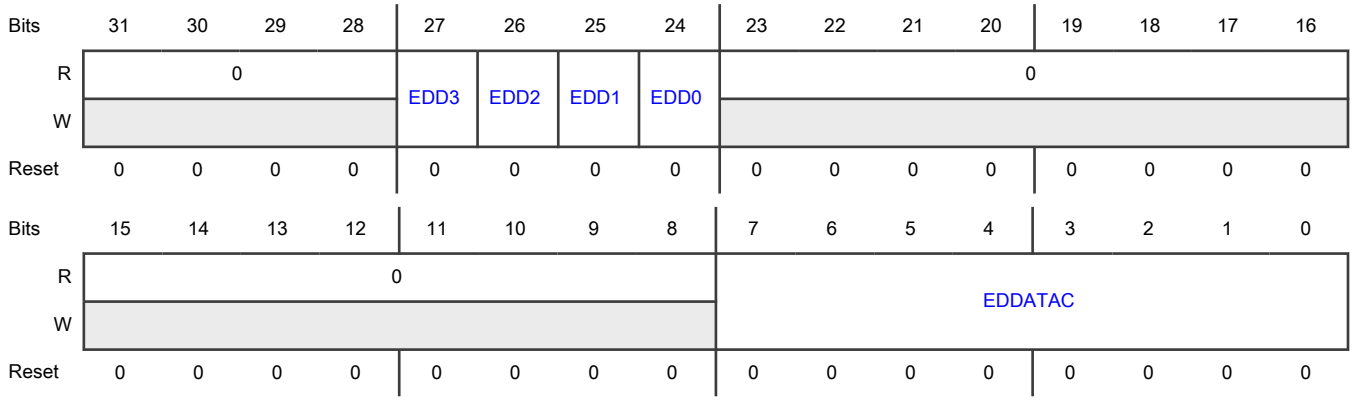
Offset

Register	Offset
UD5	E4h

Function

Provides a means to enable data to be substituted during UTest procedures.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 EDD3	EDC After ECC Logic Check Double Word 3 Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW3 on the page selected in ADR.
26 EDD2	EDC after ECC Logic Check Double Word 2 Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW2 on the page selected in ADR.
25 EDD1	EDC After ECC Logic Check Double Word 1 Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW1 on the page selected in ADR.
24 EDD0	EDC After ECC Logic Check Double Word 0 Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW0 on the page selected in ADR.
23-8 —	Reserved
7-0 EDDATAC	EDC After ECC Data Check Bits [7:0] Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field corresponds to the eight array bits representing the check bits of all double words in the page selected in ADR.

20.7.1.22 UTest Address 0 (UA0)

Offset

Register	Offset
UA0	E8h

Function

Provides a means to enable address to be inverted during UTest procedures.

NOTE

- UTest Address 0 (UA0) and UTest Address 1 (UA1) combine to represent a 52-bit wide field, which matches the width of the address internally to the flash memory that is used for the address encode comparison.
- Writing 1 to any one of the fields in UA0 or UA1 leads to flagging an AEE error when causing the error injection.
- When the value of a field within the range of bits 18:3 is 1, the corresponding address signal to the flash memory controller or BIU is inverted.

Diagram

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

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

Fields

Field	Function
31-0 AEI	Address Encode Invert [31:0] Enables checks of address encode logic by allowing address bits to be inverted into the address encode compare logic, forcing a miscompare. Performing array reads on the page selected in ADR allows the address encode invert(s) to be active.

Table continues on the next page...

Field	Function
	Additionally, in the event of an address encode error during normal user mode operation, the page address provided to the PFC is inverted when an address encode error is encountered. AEI[18:3], which reflects the page address, can be used with the Address Encode Logic Check to control this inversion feature (aligning with ADR). Any bit in AEI[18:3] written to a 1 also enables the inversion of the corresponding signal of that address, allowing checks to be performed on the comparison logic that exists in the PFC. **NOTE** Multiple bits in AEI[18:3] may be written to a 1, which causes these bits to miscompare on the address encode comparison logic, as well as causing multiple bits to have their inversion enabled as part of the address provided to the BIU.

20.7.1.23 UTest Address 1 (UA1)

Offset

Register	Offset
UA1	ECh

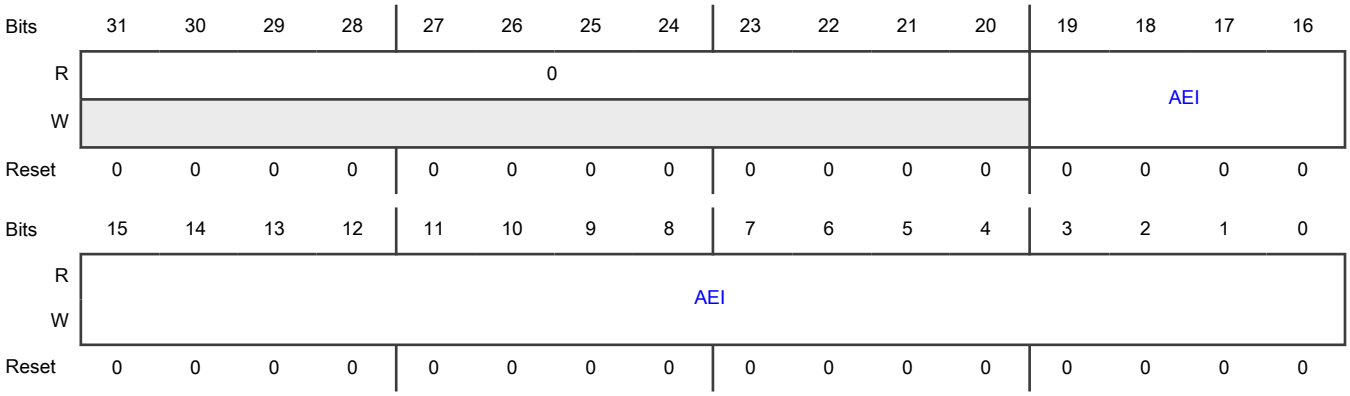
Function

Provides a means to enable address to be inverted during UTest procedures.

NOTE

- UTest Address 0 (UA0) and UTest Address 1 (UA1) combine to represent a 52-bit wide field, which matches the width of the address internally to the flash that is used for the address encode comparison.
- Writing 1 to any one of the fields in UA0 or UA1 leads to flagging an AEE error when causing the error injection.

Diagram



Fields

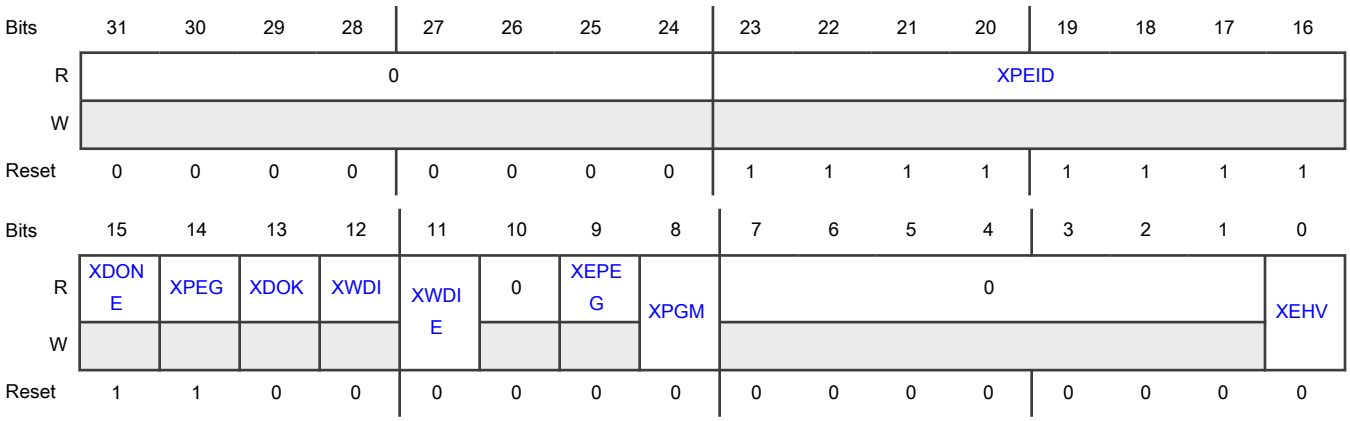
Field	Function
31-20 —	Reserved
19-0 AEI	Address Encode Invert [51:32] Enables checks of address encode logic by allowing address bits to be inverted into the address encode compare logic, forcing a miscompare. Performing array reads on the page selected in ADR allows the address encode invert(s) to be active.

20.7.1.24 Express Module Configuration (XMCR)

Offset

Register	Offset
XMCR	F0h

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 XPEID	Express Program Master/Domain ID This field shows the ID of the master that has started the express program sequence. The ID is latched when the sequence has started (writing of the XPEADR register). It is set back to the reset value (set to all '1') when the operation is completed. See the chip-specific information for a list of Master IDs.
15	Express State Machine Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
XDONE	XDONE indicates whether the embedded flash memory is performing an express program operation. XDONE is set to a 1 on completion of the express program operation. 0b - Executing an express program operation 1b - Not executing an express program operation
14 XPEG	Express Program Good The XPEG field indicates the completion status of the last flash memory express program operation. The value of XPEG is updated automatically after the program operations. XPEG will be 0 if the program operation failed. See MCRS[PEG] for more information, as the features of that bit apply to XPEG as well. 0b - Program operation failed 1b - Program operation successful
13 XDOK	Express Data OK This field is a status signal from the flash to indicate that it is OK to load Data into the DATA registers. It will assert in response to a XPEADR load (once ongoing operations are aborted, if required). XDOK will clear once the Express Program operation is complete (XPGM written to 0). 0b - Flash memory not ready to accept writes to the DATA registers 1b - Writes to DATA registers allowed
12 XWDI	Express Watch Dog Interrupt XWDI is a status register to indicate that the Watchdog Timer for Express Program had expired. XWDI is status only, and will be automatically cleared once the operation that caused the watchdog timeout is terminated or clean up from the operation is completed (clearing of XPGM bit). 0b - Normal Operation, Watchdog Timer has not expired. 1b - Express Program Watchdog Timer has expired.
11 XWDIE	Express Watch Dog Interrupt Enable XWDIE provides a mechanism to trigger an interrupt request upon the assertion of the XWDI bit on the express interface. If XWDIE is asserted, when XWDI asserts an interrupt from the flash will trigger to the system. The interrupt from the flash will mirror exactly the XWDI bit. XWDIE does not affect the register bit XWDI. Writing (and reading) of this bit will be restricted to the master that updated the XPEADR register as captured in the XPEID register at the start of the Express Program sequence. 0b - Express watchdog interrupt disabled 1b - Express watchdog interrupt enabled
10 —	Reserved
9 XEPEG	Express ECC Enabled Program Good XEPEG is a qualifier to XPEG to indicate if a passing program required ECC Enabled verifies to pass. In the event of a failed operation (XPEG=0), XEPEG will always remain 0. The value of XEPEG is updated

Table continues on the next page...

Table continued from the previous page...

Field	Function
	automatically after the program operations. See MCRS[EPEG] for more information, as the features of that bit apply to XEPEG as well. 0b - Program operation did not require ECC-enabled verifies 1b - Program operation required ECC-enabled verifies to pass
8 XPGM	Express Program XPGM is used as part of the setup for an express program operation. A 0 to 1 transition of XPGM after program interlock write(s) is part of the setup for an express program sequence. A 1 to 0 transition of XPGM ends the express program sequence. XPGM can be set at any time if UT0[UTE] is low, and after the interlock write is completed (XPEADR written, XDOK asserted, and DATAn register(s) written). XPGM may be cleared when XEHV is low. Writing (and reading) of this bit will be restricted to the master that updated the XPEADR register as captured in the XPEID register at the start of the Program sequence. 0b - Flash memory not executing an express program sequence 1b - Flash memory executing an express program sequence
7-1 —	Reserved
0 XEHV	Express Enable High Voltage The XEHV field enables the embedded flash memory for a high voltage program/erase operation. XEHV is cleared on reset. XEHV must be set after DATA writes to start an express program operation and XPGM is set. This field may be set when XPGM = 1, XDOK = 1, and XWDI = 0. XEHV may not be written to a 1 after it is cleared as part of an express program cleanup, until after the next XPEADR write. Express Program operations may not be aborted. XEHV may only be cleared once XDONE is set to 1. Writing (and reading) of this bit will be restricted to the master that updated the XPEADR register as captured in the XPEID register at the start of the Program sequence. 0b - Flash memory is not enabled to perform an express high voltage operation. 1b - Flash memory is enabled to perform an express high voltage operation.

20.7.1.25 Express Program Address (XPEADR)

Offset

Register	Offset
XPEADR	F4h

Function

The Express Program Address register (XPEADR) is used to provide the address to be programmed. The express program address register is read only in user mode (user mode writes occur with writes done to the Platform Flash Controller). The address is identified using a combination of bits that identify the memory region and offset address.

NOTE

- The address given is an offset from the base address of the address space.
- The block numbering scheme that corresponds to the offset address is based on the flash memory's *internal* addressing scheme, which may be different than the chip's system addressing scheme.

Understanding the mapping between the flash address, which is specified relative to the flash module's internal addressing scheme, and the location within the chip's system memory map requires a clear understanding of the flash memory layout. See the chip-specific information for a detailed explanation of the chip's flash memory layout and how to map system addresses to the flash module's internal addressing.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							XPEA5	XPEA4	XPEA3	XPEA2	XPEA1	XPEA0	XPEADDR		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31-25 —	Reserved Reserved; reset to 0
24 XPEA5	Express Program Address Region 5 Qualifies the address field (XPEADDR) to the region. If XPEA5 = 1, the XPEADDR field maps to that region. See the description of the XPEADDR field for more information. If this region is not present, the field is locked to 0. 0b - Address accessed is not from region 5 1b - Address accessed is from region 5
23 XPEA4	Express Program Address Region 4 Qualifies the address field (XPEADDR) to the region. If XPEA4 = 1, the XPEADDR field maps to that region. See the description of the XPEADDR field for more information. If this region is not present, the field is locked to 0. 0b - Address accessed is not from region 4 1b - Address accessed is from region 4
22 XPEA3	Express Program Address Region 3 Qualifies the address field (XPEADDR) to the region.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If XPEA3 = 1, the XPEADDR field maps to that region. See the description of the XPEADDR field for more information. If this region is not present, the field is locked to 0. 0b - Address accessed is not from region 3 1b - Address accessed is from region 3
21 XPEA2	Express Program Address Region 2 Qualifies the address field (XPEADDR) to the region. If XPEA2 = 1, the XPEADDR field maps to that region. See the description of the XPEADDR field for more information. If this region is not present, the field is locked to 0. 0b - Address accessed is not from region 2 1b - Address accessed is from region 2
20 XPEA1	Express Program Address Region 1 Qualifies the address field (XPEADDR) to the region. If XPEA1 = 1, the XPEADDR field maps to that region. See the description of the XPEADDR field for more information. If this region is not present, the field is locked to 0. 0b - Address accessed is not from region 1 1b - Address accessed is from region 1
19 XPEA0	Express Program Address Region 0 Qualifies the address field (XPEADDR) to the region. If XPEA0 = 1, the PEADDR field maps to that region. See the description of the XPEADDR field for more information. If this region is not present, the field is locked to 0. 0b - Address accessed is not from region 0 1b - Address accessed is from region 0
18-5 XPEADDR	Express Program Address The XPEADDR register provides offset address location to be programmed. This address is always a quad page address that selects 1024 bits. The XPEADDR register is read only in user mode and represents the flash physical address status. XPEADDR may be updated once (and only once) per Express Program event. XPEADDR writes initiate an Express Program request. XPEADDR may be updated four times between reset events. Upon completing the fourth Express Program request (clearing of XPGM register), the XPEADDR register will return to all zeros, and remain locked from future XPEADDR updates and thus future Express Program requests will be ignored until the next reset. Once XPEADDR is updated, it will be locked for the full express program operation, until XMCR[XPGM] is cleared at the end of the operation. Writing of XPEADDR has the effect of blocking other registers used for program and erase to be written (PEADR/MCR registers as well as APEADR/APDATA/AMCR registers) until the express program operation is completed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Upon completion of an Express Program operation (XMCR[XPGM] written to 0) the XPEADR registers will return to an all zeros state. When UT0[UTE] is set or XMCR[XEHV] is set, XPEADDR bits will not be writeable. Once XPEADDR is updated, it will have read restrictions based on the PEID master that did the original write in the Platform Flash Controller. XPEADDR[18:5] is an offset from a base address of 0h for each block region. The XPEA0, XPEA1, XPEA2, XPEA3, XPEA4 and XPEA5 qualify the block size region the XPEADDR field.
4-0 —	Reserved Reserved; reset to 0

20.7.1.26 Program Data (DATA0 - DATA31)

Offset

For a = 0 to 31:

Register	Offset
DATAa	100h + (a × 4h)

Function

Enables data to be provided for program (or express program). This set of 32 data registers enables 1024 bits of data to be programmed.

Diagram

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

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

Fields

Field	Function
31-0	Program Data

Table continues on the next page...

Field	Function
PDATA	PDATA registers will reset to all ones. PDATA is not writable before PEADR is written or updated to start a program, express program, erase or UTest operation, and will read a value of all ones. Once PEADR is written, all PDATA registers will be made available for writing. Writing (and reading) of the PDATA registers will be restricted to the master that updated the PEADR register as captured in the PEID register. **NOTE** PDATA restrictions related to PEID will be disabled if UT0[UTE] is set. **NOTE** For Express Program a similar behavior exists on the PDATA registers. Once XPEADR is written, and XDOK comes back as a 1, all PDATA registers will switch to all 1's (program safe state) and be made available for writing (and reading) to the master's ID that matches XPEID captured during the XPEADR update. In the event of an interrupted operation, XPEID permissions will override PEID permissions until the express program operation is completed (XPGM written to a 0). Upon completion of an operation (Program (MCR[PGM] written to 0 and MCRS[PES] equals 0), Erase (MCR[ERS] written to 0 and MCRS[PES] equals 0), Express Program (XMCR[XPGM] written to 0), Array Integrity (UT0[AIE] written to 0), User Margin Read(UT0[AIE] written to 0)) or Program and Erase Sequence clear (MCRS[PES] written to 0) the PDATA registers will return to an all ones state. When UT0[AIE] is set, PDATA bits will not be writeable. When MCRS[PES] is set, PDATA bits will not be writeable except when an express program request occurs and XMCR[XDOK] is set. Also, when MCR[PGM] or MCR[ERS] are set, PDATA bits will not be writeable except when an express program request occurs and XMCR[XDOK] is set. When an express program operation is in process, PDATA bits will not be writeable once XMCR[XEHV] is set until XMCR[XPGM] is cleared. Only PDATA Bits that are written will receive program or express program pulses.

20.8 Glossary

Abort Premature end of a mode, sequence, state or operation. An abort may leave the embedded flash memory in a state in which it contains indeterminate or corrupted data.

BIU Bus interface unit(Contains all system-level customization required to make the embedded flash memory part of an SoC)

Block Subdivision of the Flash Module containing NVM memory bits. Block sizes range from 256KB to 2MB. Each block is made up of 8KB sectors.

Double word Two words, or 64 bits.

ECC Error correction code(Internally used to correct single bit errors, or detect double errors within a 64-bit double word.)

ECC segment 64 bit double word, representing the size of data used to calculate ECC information.

EDC Error detection code

FC Flash memory core

FMU	Flash Management Unit - Logic that enables program, erase, read, and other flash functions.
Interlock Write	A write to the PEADR register followed by a write to any DATA register performed while initiating a program, erase, array integrity, or user margin read sequence. Interlock in cleared once the operation is finished by clearing the appropriate control bit, causing PEADR register to clear.
MCU	Microcontroller unit
Module	Flash memory instance that contains flash block(s), FMU, and required analog support structures.
NVM	Nonvolatile memory
OTP	One time programmable
PFC	Platform flash memory controller(Contains all system-level customization required to make the embedded flash memory part of an SoC)
Page	8 words of data (256 bits). When doing reads to the embedded flash memory, this is the width of data read. Up to 4 pages can be programmed at a time.
Partition	Subdivision of the Flash Module. Used for read-while-write operation, where addresses in the partition being written may not be read. Partitions is always equal to 1 block, and contains multiple sectors.
RWW	Read-while-write (operation)
Sector	Subdivision of the Flash Block that is independently erasable. Sector Size is always 8 KB.
Super sector	Subdivision of the flash block that includes a group of sectors. Super Sector Size is always 64 KB, and consists of 8 sectors.
SoC	System-on-chip
UTest NVM sector	8 KB sector outside the main address space
Word	32 bits

Chapter 21
Flash Memory Controller (PFLASH)

21.1 Chip-specific PFLASH information

21.1.1 Flash memory architecture

The flash memory on the chip consists of a flash memory controller and a flash memory array module. The flash memory controller provides flash–memory configuration and control functions and manages the interface between the flash memory array and the chip's crossbar switch.

This chip implements upto four 64-bit AHB buses.

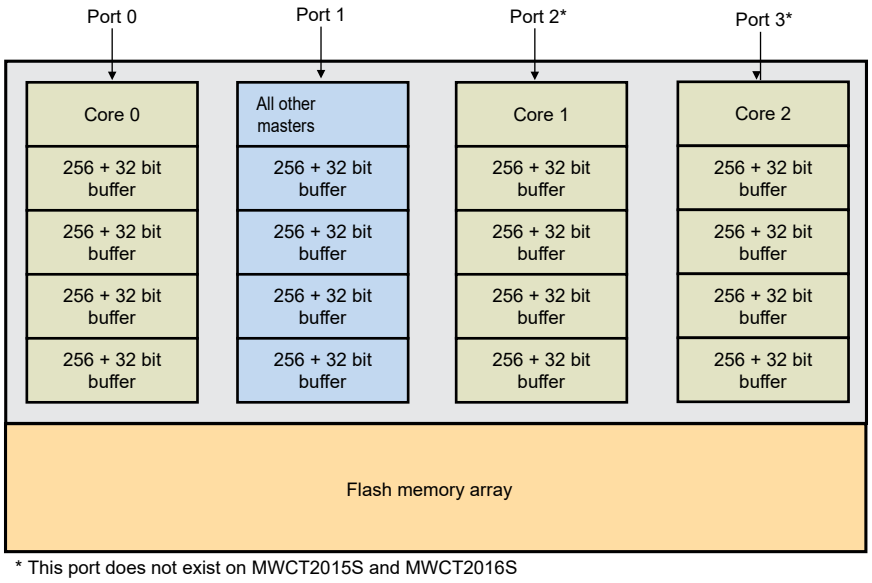


Figure 49. Flash memory architecture

21.1.2 Flash memory controller

The flash memory controller:

- Acts as an interface between the system bus and flash memory array.
- Is a triple-ported controller with a dedicated line buffering per port and per master ID. This enables you to use the line buffers more efficiently because various masters have dedicated buffers that are not compromised when other masters perform read operations.

Also, by having separate ports, you can have separate connections for each CPU instruction bus and a single port for all data accesses as shown in Figure 49.

In general, the flash memory controller registers affect the global flash memory behavior (for example, read buffering and access control).

NOTE

In MWCT2016S:

- PFLASH block 3 and block 4 are not present, so both the sector and super sector registers are not available.
- For PFLASH block 2, the super sector registers are not available.

21.1.3 Platform flash configuration registers (PFCR*n*)

The table below defines the PFCR*n* fields that the chip uses.

Table 90. Platform flash configuration registers (PFCR*n*)

Master	Buffer enable field	Prefetch enable field
Cortex–M7_0	P0_CBFEN, P0_DBFEN	P0_CPFEN, P0_DPFEN
eDMA + HSE_B + EMAC[1] + GMAC_1	P1_CBFEN, P1_DBFEN	P1_CPFEN, P1_DPFEN
Cortex–M7_1	P2_CBFEN, P2_DBFEN	P2_CPFEN, P2_DPFEN
Cortex–M7_2	P3_CBFEN, P3_DBFEN	P3_CPFEN, P3_DPFEN

1. MWCT2015S and MWCT2016S do not support EMAC.

21.1.4 Platform flash access protection register (PFAPR)

This table defines the PFAPR[M*n*AP] fields that the chip uses. This chip does not use the other master access protection fields, but those fields are readable and writable.

Table 91. Platform flash access protection register (PFAPR)

Master number	Master name	Access protection field
0	Cortex–M7_0 AHBM	M0AP
1	eDMA	M1AP
2	HSE_B	M2AP
3	EMAC	M3AP
4	Cortex–M7_1 AHBM	M4AP
5	Cortex–M7_2 AHBM	M5AP
6	uSDHC/ GMAC_1	M6AP
7	Cortex–M7_3 AHBM	M7AP

21.2 Introduction

PFLASH acts as an interface between the system bus (AHB-Lite 2.v6) and flash memory array.

PFLASH supports three 64-bit AHB buses and a 256-bit read data interface from each flash memory array. The slave port assignments and buffer organization are organized to offer maximum performance of code execution in a multicore architecture. The buffer mechanism serves to deliver flash memory read data with zero-wait state response on lines that reside in cache. AHB requests that miss the prefetch cache generate the needed flash memory array access and are forwarded to the AHB upon completion. Each cache entry is 256 bits, matching the flash memory array page size and providing 512 bytes of high-speed local storage.

21.3 Features

- Three 64-bit AHB interface ports (p0, p1, p2) allowing simultaneous access to dedicated prefetch mini-cache per slave port
- 256-bit read data bus and 64-bit write data bus
- Configurable read buffering and line prefetching support via a mini-cache, plus a prefetch controller for each AHB port to provide single-cycle buffer hit read response
- Configurable access control based on read/write and AHB master ID attributes

- Support for reporting single-bit and multi-bit flash memory ECC events on a 64-bit doubleword boundary

21.4 Block diagrams

The following figure provides a block diagram showing PFLASH and the attached flash memory array.

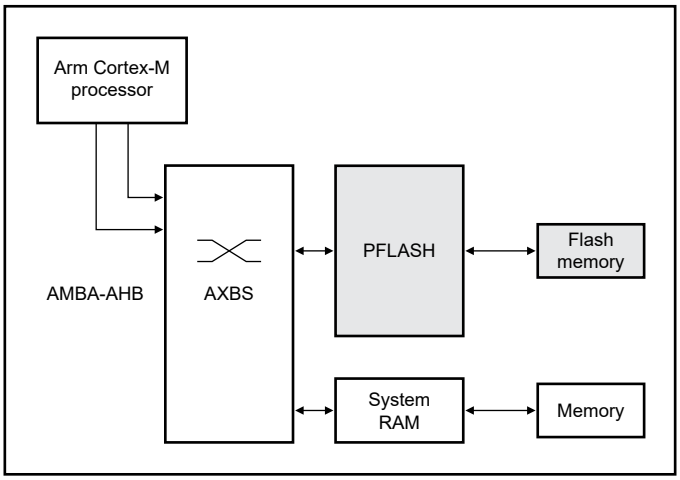


Figure 50. Platform-centric simple block diagram with PFLASH

21.5 Functional description

As shown in Figure 50, PFLASH interfaces between:

- The AHB system bus ports
- The flash memory array

For accesses targeting flash memory, the PFLASH generates as inputs to the flash memory array:

- Read and write enables
- Block selects
- Array address
- Write size
- Write data

PFLASH captures read data from the flash memory array and drives it onto the AHB system bus. Up to four pages of data (256-bit page size) may be buffered in each prefetch buffer for AHB Port0, Port1, and Port2. Lines may be prefetched in advance of being requested, allowing single-cycle (zero AHB wait-states) read data responses on buffer hits.

Access protections may be applied on a per-master basis for both reads and writes to support security and privilege mechanisms.

21.5.1 Read transactions

On an incoming AHB read request, a mini-cache lookup and access privilege evaluation are performed during the AHB address phase. If a buffer hit occurs, the requested data is retrieved from the previously loaded prefetch buffer entry and returned on the system bus with a zero wait-state response. If a buffer miss occurs, a flash memory access is initiated.

Read accesses are terminated under control of the appropriate wait state settings. Access timing can be varied to account for the operating conditions of the chip (for example, frequency, voltage, temperature, and so on) by appropriately setting the read wait state field in flash memory.

21.5.2 Write transactions

An interlock write on a program or erase sequence is initiated by first writing to Platform Flash Memory Program Erase Address Logical (PFCPGM_PEADR_L), Platform Flash Memory Express Program Erase Address Logical (PFCPGM_XPEADR_L)(see the Flash Memory chapter for write sequence details).

21.5.3 Access protections

21.5.3.1 PFAPR

PFLASH provides programmable, configurable access protections for read cycles on a per-master basis in PFAPR[M*n*AP]. This field restricts read access on a per-master basis. This functionality is described in Platform Flash Memory Access Protection (PFAPR). Detection of a protection violation based on PFAPR settings results in an error response from PFLASH on the AHB transfer.

21.5.4 Error termination

PFLASH can invoke a system bus error termination in the following scenarios:

- Access privilege violation (see Access protections for details)
- Attempted access by an AHB master to a reserved region in the flash memory map
- Multi-bit ECC error detection on AHB read, and PFCR4[DMEEE] = 0 (for data flash memory, it is further qualified by PFCR4[DERR_SUP] = 0)

21.5.5 Line read buffers and prefetch operation

The PFLASH AHB ports of the each contain a mini-cache. PFLASH uses the buffers for both prefetch and normal demand fetches. Also, the buffers are shared for code and data fetches, and can be controlled independently for code and data from control registers.

Prefetch triggering is controllable on a per-port basis. A PFLASH read access may trigger a prefetch to the next sequential line of array data on the cycle following the request. The access address is incremented by 32 bytes, and a subsequent flash memory access is initiated. A flash memory array prefetch is initiated if the data is not already resident in a line read buffer. Prefetched data is always loaded into the least-recently-used buffer.

For Port0, Port1, and Port2 there are four line buffer entries in their respective prefetch mini-caches that follow a fully associative, least-recently-used replacement policy.

For prefetching to occur, you must set the following configuration fields, where *n* corresponds to the port number and *m* corresponds to *C* (code) or *D* (data):

PFCRn[Pn_mBFEN] = 1 and PFCRn[Pn_mPFEN] = 1

21.5.6 Array integrity considerations

During an array integrity sequence, the flash memory array ignores any incoming read requests. When a flash memory array integrity check is in progress, PFLASH terminates all flash memory access requests with an error. More specifically, it aborts the incoming flash memory access requests and terminates the system bus transfer with an error.

21.5.7 Safety considerations

21.5.7.1 Flash memory address generation check

Functional safety coverage of the address path and control within PFLASH rely on a feedback path between PFLASH and flash memory. Remember that on a requested access to flash memory, PFLASH must decode the system AHB bus signals to generate the corresponding flash memory interface signals to invoke a flash memory lookup. In addition to providing the requested read data, the flash memory also provides output sidebands reflecting the encoded address and block selects used to perform the actual row lookup.

PFLASH uses this sideband information to verify the expected transaction. If a mismatch is detected, indicating a failure in the address generation or control logic within PFLASH or the transmission path between PFLASH and the flash memory array, then the event is forwarded to the chip fault collection module and the corresponding buffer is invalidated.

21.5.8 ECC error handling on data flash block

When PFCR4[DERR_SUP] is enabled, ECC errors on data flash blocks are handled specially.

If there is a noncorrectable error detection, a fixed, illegal opcode value is returned to the requesting master along with the associated ECC checkbits as determined by the requesting address.

For noncorrectable error detection, PFLASH returns a value of 1555_1555h to the requesting master.

This is mainly used for EEPROM emulation applications.

21.5.9 Read cycles—buffer miss

On an incoming AHB read request, a mini-cache lookup and access privilege evaluation are performed during the AHB address phase. If a buffer miss occurs, a flash memory access is initiated.

If the flash memory access was the direct result of an AHB transaction, the corresponding page buffer is loaded and marked as the most-recently-used. If the flash memory access was the result of a speculative prefetch to the next sequential line, it is loaded into the least-recently-used buffer. The status of this buffer is not changed to most-recently-used until a subsequent buffer hit occurs as a result of an AHB read request.

21.5.10 Read cycles—buffer hit

PFLASH allows single-cycle read responses to the AHB when the requested read access was previously loaded into one of the page buffers. In these cases of a buffer hit, read data is returned on the system bus with a zero wait-state response.

21.5.11 Flash memory error response operation

The flash memory array may signal an error response to terminate a requested access because of improper sequencing during program/erase operations and improper sequencing during array integrity testing. When an error response is received, PFLASH does not update or validate a page read buffer. An error response may be signaled on a read or interlock write operation. For more information on the specifics related to signaling of flash memory errors, including flash memory ECC events, array integrity testing, and read-while-write events, see the flash memory chapter.

21.6 PFLASH register descriptions

PFLASH provides an IPS programming model mapped to a standard 16 KB on-platform peripheral slot. The programming model consists of flash memory access configuration.

You can reference the programming model only by using a 32-bit (word) access. References that are attempted using different access sizes, or to undefined (reserved) addresses, or in User mode generate an IPS error termination. PFLASH allows access to the programming model by all system bus masters.

Write to read only registers don't generate error termination.

You can only access the programming model in Supervisor mode, except *PEADR* registers which can be accessed in Supervisor or User mode.

Attempted updates to the programming model when PFLASH is in the middle of an operation results in non-deterministic behavior. You must architect software to avoid this scenario. The recommended flow for multicore devices is:

1. Start only one core.
2. Execute initialization code until it is complete.
3. Start the remaining cores.

If you need to reconfigure the flash memory, code execution must be temporarily moved to system RAM.

21.6.1 PFLASH memory map

PFLASH base address: 4026_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Platform Flash Memory Configuration 0 (PFCR0)	32	RW	0000_0003h
4h	Platform Flash Memory Configuration 1 (PFCR1)	32	RW	0000_0003h
8h	Platform Flash Memory Configuration 2 (PFCR2)	32	RW	0000_0003h
10h	Platform Flash Memory Configuration 4 (PFCR4)	32	RW	0000_0000h
14h	Platform Flash Memory Access Protection (PFAPR)	32	RW	FFFF_FFFFh
300h	Platform Flash Memory Program Erase Address Logical (PFCPGM_PEADR_L)	32	RW	0000_0000h
304h	Platform Flash Memory Program Erase Address Physical (PFCPGM_PEADR_P)	32	R	0000_0000h
308h	Platform Flash Memory Express Program Erase Address Logical (PFCPGM_XPEADR_L)	32	RW	0000_0000h
30Ch	Platform Flash Memory Express Program Erase Address Physical (PFCPGM_XPEADR_P)	32	R	0000_0000h
340h - 350h	Block n Sector Program Erase Lock (PFCBLK0_SPELOCK - PFCBLK4_SPELOCK)	32	RW	FFFF_FFFFh
358h	Block UTEST Sector Program Erase Lock (PFCBLKU_SPELOCK)	32	RW	0000_0001h
35Ch - 368h	Block n Super Sector Program Erase Lock (PFCBLK0_SSPELOCK - PFCBLK3_SSPELOCK)	32	RW	0000_0FFFh
380h - 390h	Block n Set Sector Lock (PFCBLK0_SETSLOCK - PFCBLK4_SETSLOCK)	32	RW	0000_0000h
398h	Block UTEST Set Sector Lock (PFCBLKU_SETSLOCK)	32	RW	0000_0000h
39Ch - 3A8h	Block n Set Super Sector Lock (PFCBLK0_SSETSLOCK - PFCBLK3_SSETSLOCK)	32	RW	0000_0000h
3C0h - 45Ch	Block a Lock Master Sector b (PFCBLK0_LOCKMASTER_S0 - PFCBLK4_LOCKMASTER_S7)	32	R	FFFF_FFFFh
480h	Block UTEST Lock Master Sector (PFCBLKU_LOCKMASTER_S)	32	R	0000_00FFh
484h	Block m Lock Master Super Sector n (PFCBLK0_LOCKMASTER_SS0)	32	R	FFFF_FFFFh
488h	Block m Lock Master Super Sector n (PFCBLK0_LOCKMASTER_SS1)	32	R	FFFF_FFFFh
48Ch	Block m Lock Master Super Sector n (PFCBLK0_LOCKMASTER_SS2)	32	R	FFFF_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
494h	Block m Lock Master Super Sector n (PFCBLK1_LOCKMASTER_SS0)	32	R	FFFF_FFFFh
498h	Block m Lock Master Super Sector n (PFCBLK1_LOCKMASTER_SS1)	32	R	FFFF_FFFFh
49Ch	Block m Lock Master Super Sector n (PFCBLK1_LOCKMASTER_SS2)	32	R	FFFF_FFFFh
4A4h	Block m Lock Master Super Sector n (PFCBLK2_LOCKMASTER_SS0)	32	R	FFFF_FFFFh
4A8h	Block m Lock Master Super Sector n (PFCBLK2_LOCKMASTER_SS1)	32	R	FFFF_FFFFh
4ACh	Block m Lock Master Super Sector n (PFCBLK2_LOCKMASTER_SS2)	32	R	FFFF_FFFFh
4B4h	Block m Lock Master Super Sector n (PFCBLK3_LOCKMASTER_SS0)	32	R	FFFF_FFFFh
4B8h	Block m Lock Master Super Sector n (PFCBLK3_LOCKMASTER_SS1)	32	R	FFFF_FFFFh
4BCh	Block m Lock Master Super Sector n (PFCBLK3_LOCKMASTER_SS2)	32	R	FFFF_FFFFh

21.6.2 Platform Flash Memory Configuration 0 (PFCR0)

Offset

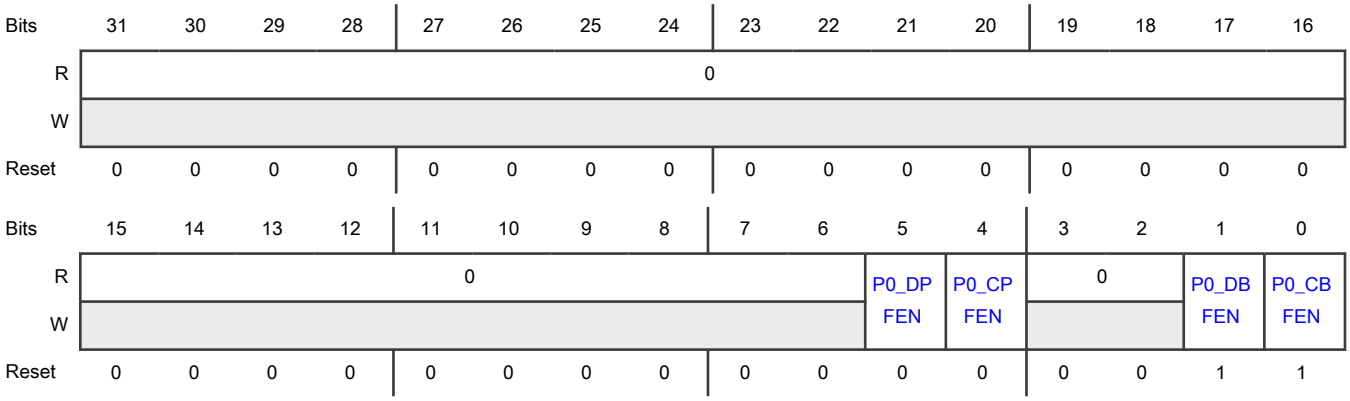
Register	Offset
PFCR0	0h

Function

Specifies the operation of PFLASH Port0.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 P0_DPFEN	Port0 Data Prefetch Enable Enables or disables data prefetching initiated by a read access on Port0. Prefetching can only be enabled if the buffers are enabled by writing 1 to DBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
4 P0_CPFEN	Port0 Code Prefetch Enable Enables or disables code prefetching initiated by a read access on Port0. Prefetching can only be enabled if the buffers are enabled by writing 1 to CBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
3-2 —	Reserved
1 P0_DBFEN	Port0 PFLASH Line Read Data Buffers Enable Enables or disables line read data buffer hits. It is also used to invalidate the buffers. If this field is 0, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If this field is enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable
0 P0_CBFEN	Port0 PFLASH Line Read Code Buffers Enable Enables or disables line read code buffer hits. It is also used to invalidate the buffers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If disabled, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable

21.6.3 Platform Flash Memory Configuration 1 (PFCR1)

Offset

Register	Offset
PFCR1	4h

Function

Specifies the operation of PFLASH Port1.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										P1_DPFEN	P1_CPFEN	0		P1_DBFEN	P1_CBFEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-6 —	Reserved
5 P1_DPFEN	Port1 Data Prefetch Enable Enables or disables data prefetching initiated by a read access on Port1. Prefetching can only be enabled if the buffers are enabled by writing 1 to DBFEN. Hardware reset returns this field to 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
4 P1_CPFEN	Port1 Code Prefetch Enable Enables or disables code prefetching initiated by a read access on Port1. Prefetching can only be enabled if the buffers are enabled by writing 1 to CBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
3-2 —	Reserved
1 P1_DBFEN	Port1 PFLASH Line Read Data Buffers Enable Enables or disables line read data buffer hits. It is also used to invalidate the buffers. If this field is 0, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If this field is enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable
0 P1_CBFEN	Port1 PFLASH Line Read Code Buffers Enable Enables or disables line read code buffer hits. It is also used to invalidate the buffers. If disabled, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable

21.6.4 Platform Flash Memory Configuration 2 (PFCR2)

Offset

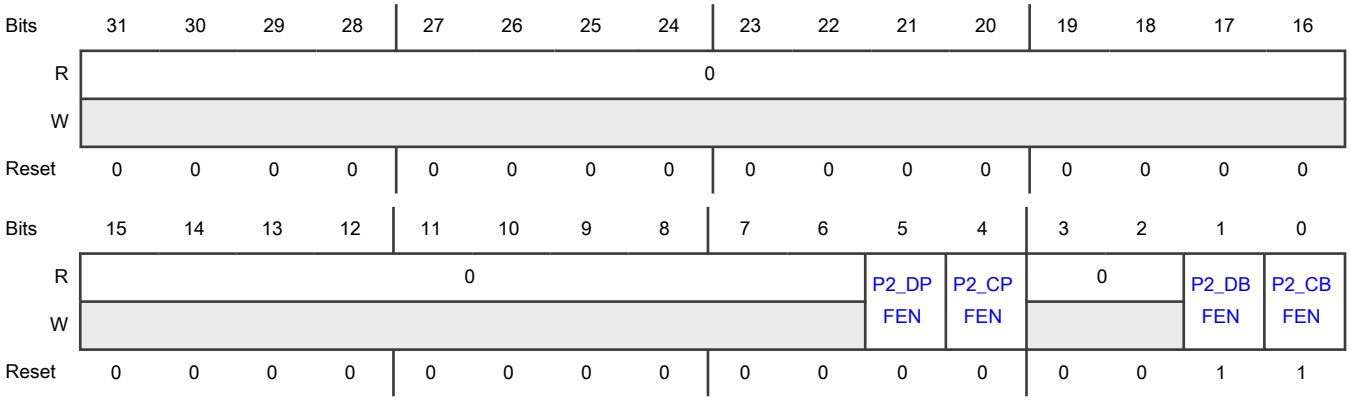
Register	Offset
PFCR2	8h

Function

Specifies the operation of PFLASH Port2.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 P2_DPFEN	Port2 Data Prefetch Enable Enables or disables data prefetching initiated by a read access on Port2. Prefetching can only be enabled if the buffers are enabled by writing 1 to DBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
4 P2_CPFEN	Port2 Code Prefetch Enable Enables or disables code prefetching initiated by a read access on Port2. Prefetching can only be enabled if the buffers are enabled by writing 1 to CBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
3-2 —	Reserved
1 P2_DBFEN	Port2 PFLASH Line Read Data Buffers Enable Enables or disables line read data buffer hits. It is also used to invalidate the buffers. If this field is 0, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If this field is enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable
0 P2_CBFEN	Port2 PFLASH Line Read Code Buffers Enable Enables or disables line read code buffer hits. It is also used to invalidate the buffers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If disabled, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable

21.6.5 Platform Flash Memory Configuration 4 (PFCR4)

Offset

Register	Offset
PFCR4	10h

Function

Specifies operation of the flash memory controller buffer.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								DMEEE	0			BLK4_PS			DERR_SUP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7 DMEEE	Disable Multi-Bit ECC Error Exception Enables or disables system bus error response on multi-bit ECC error. Hardware reset returns this field to 0. 0b - Error response sent on system bus for multi-bit ECC error 1b - Error response not sent on system bus for multi-bit ECC error

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-4 —	Reserved
3-1 BLK4_PS	Block 4 Pipe Select Selects the active pipe for flash memory block 4 access. PFLASH has four independent command pipes to issue four parallel read commands to different flash memory blocks. Reads from flash memory block 0–3 are always done through command pipe 0–3, respectively. However, the access to block 4 is not fixed and can be through any of the command pipes. You must only change this field when there is no ongoing block 4 access. A special round-robin arbitration scheme snoops the availability of a command pipe during block 4 access. If any of the command pipes are idle during the first read request to block 4, the ownership of that command pipe gets shared between block 4 and the respective block. If none of the command pipes are idle during a block 4 read request, block 4 gets associated with each of the command pipes in round-robin fashion. When a command pipe acquires ownership of block 4, it keeps that ownership until all the commands to block 4 from all the masters are served. 000b - Block 4 access is always through pipe0 001b - Block 4 access is always through pipe1 010b - Block 4 access is always through pipe2 011b - Block 4 access is always through pipe3 1xxb - Block 4 access can be through any of the command pipes, based on which command pipe is available for block 4 access
0 DERR_SUP	Data Error Suppression See the Embedded Flash Memory configuration information or system memory map for which flash memory blocks are affected by this field. 0b - Reports ECC events on data flash memory accesses 1b - Single-bit and multi-bit ECC events on data flash memory accesses are suppressed

21.6.6 Platform Flash Memory Access Protection (PFAPR)

Offset

Register	Offset
PFAPR	14h

Function

Controls read accesses to the flash memory array.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M0AP		M1AP		M2AP		Reserved		M4AP		M5AP		M6AP		M7AP	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M8AP		M9AP		M10AP		M11AP		M12AP		M13AP		M14AP		M15AP	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30: M0AP 29-28: M1AP 27-26: M2AP 25-24: — 23-22: M4AP 21-20: M5AP 19-18: M6AP 17-16: M7AP 15-14: M8AP 13-12: M9AP 11-10: M10AP 9-8: M11AP 7-6: M12AP 5-4: M13AP 3-2: M14AP 1-0: M15AP	Master n Access Protection Controls whether read accesses to the flash memory are allowed based on the master ID of a requesting master. These fields are initialized by hardware reset. The location that would be M3AP is a reserved field. x0b - This master cannot perform any read accesses x1b - This master can perform read accesses

21.6.7 Platform Flash Memory Program Erase Address Logical (PFCPGM_PEADR_L)

Offset

Register	Offset
PFCPGM_PEADR_L	300h

Function

Provides the flash memory address to be programmed, or the location of the sector or block to be erased through main flash memory (pgm/erase) queue. Write access to this register is domain-ID aware.

The respective bus master must have program/erase permission to the flash memory address written to this register. Otherwise a transfer error results. For further information on flash memory address restrictions see the XRDC chapter.

A write to this register is managed via three-cycle access. Before updating the register, you must ensure that no ongoing high-voltage operation is executing through the flash memory main queue.

Unauthorized flash memory address writes result in a transfer error.

Writes to this register during an ongoing high-voltage operation (initiated through the flash memory main queue) or during express program operation are ignored.

Diagram

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

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

Fields

Field	Function
31-0 PEADR_L	Program Erase Address Logical Contains the system logical address for flash memory program/erase.

21.6.8 Platform Flash Memory Program Erase Address Physical (PFCPGM_PEADR_P)

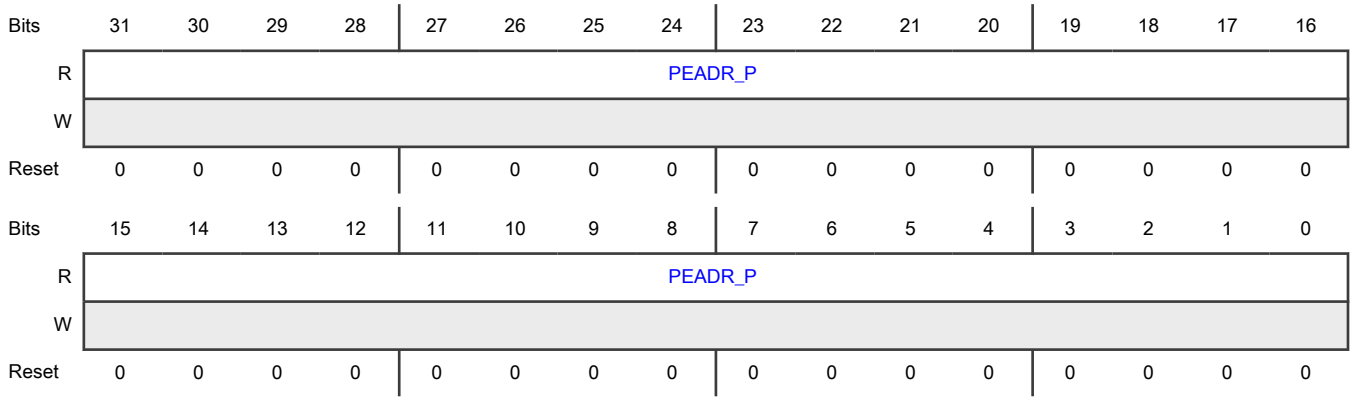
Offset

Register	Offset
PFCPGM_PEADR_P	304h

Function

Reflects the flash memory block number and offset address corresponding to Platform Flash Memory Program Erase Address Logical (PFCPGM_PEADR_L). This register has the same format as the PEADR register in the Flash Memory chapter—see it for details.

Diagram



Fields

Field	Function
31-0 PEADR_P	Program Erase Address Physical Contains the flash block select and offset address for flash memory program/erase.

21.6.9 Platform Flash Memory Express Program Erase Address Logical (PFCPGM_XPEADR_L)

Offset

Register	Offset
PFCPGM_XPEADR_L	308h

Function

Provides the flash memory address to be programmed using the flash memory express program feature. Write access to this register is domain-ID aware.

The respective bus master must have program/erase permission to the flash memory address written to this register. Otherwise a transfer error results. See the XRDC chapter for further information on flash memory address restrictions.

A write to this register is managed via three-cycle access. Before updating the register, you must ensure that no ongoing high-voltage operation is executing through the flash memory main queue.

Unauthorized flash memory address writes result in a transfer error.

Writes to this register during an ongoing express program operation are ignored.

Diagram

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

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

Fields

Field	Function
31-0 XPEADR_L	Express Program Erase Address Logical Contains the system logical address for express flash program/erase.

21.6.10 Platform Flash Memory Express Program Erase Address Physical (PFCPGM_XPEADR_P)

Offset

Register	Offset
PFCPGM_XPEADR_P	30Ch

Function

Reflects the flash memory block number and offset address corresponding to PFCPGM_XPEADR_L. This register has the same format as the XPEADR register in the Flash Memory chapter—see it for details.

Diagram

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

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

Fields

Field	Function
31-0 XPEADR_P	Express Program Erase Address Physical Contains the flash memory block select and offset address for flash memory express program/erase.

21.6.11 Block n Sector Program Erase Lock (PFCBLK0_SPELOCK - PFCBLK4_SPELOCK)

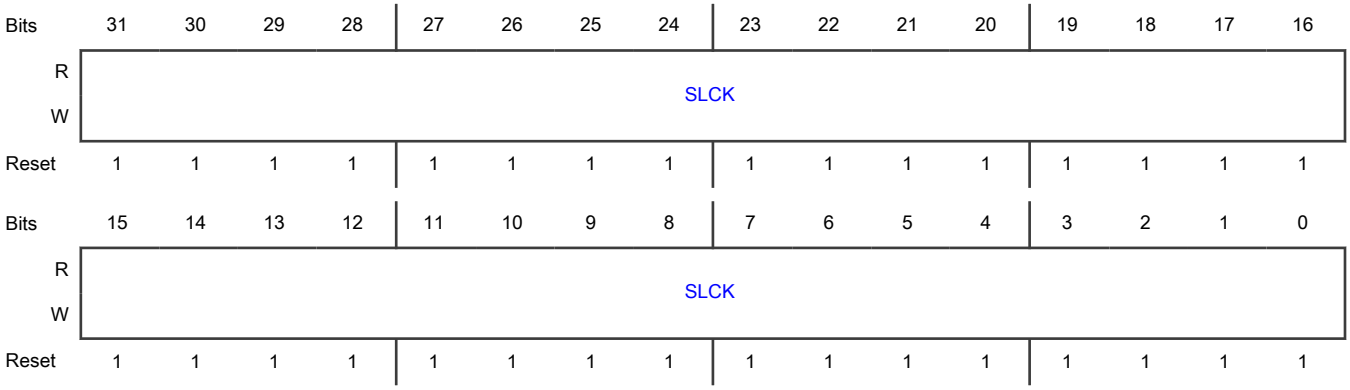
Offset

Register	Offset
PFCBLK0_SPELOCK	340h
PFCBLK1_SPELOCK	344h
PFCBLK2_SPELOCK	348h
PFCBLK3_SPELOCK	34Ch
PFCBLK4_SPELOCK	350h

Function

Provides a way to protect sectors from being modified. Sector protection is available on the last 256 KB of every block (for 256 KB blocks, all sectors are available for protection). Each lock bit can be associated with a particular domain ID by writing 1 to the appropriate bit in PFCBLKn_SETSLOCK[SETSLCK]. After the lock is acquired, only a master having the same domain ID can modify the corresponding lock bit. If the corresponding PFCBLKn_SETSLOCK[SETSLCK] bit is not equal to 1, any master can modify the appropriate SLCK bit. If a lock bit is already acquired by a particular domain ID, any effort to modify (1 to 0, or 0 to 1) the lock bit by a master with a different domain ID results in a transfer error.

Diagram



Fields

Field	Function
31-0	Sector Lock

Table continues on the next page...

Field	Function
SLCK	Locks selected sector. If vector bit value = 0, the sector is available for program and erase operations. If vector bit value = 1, the sector is locked and not available for program and erase operations.

21.6.12 Block UTEST Sector Program Erase Lock (PFCBLKU_SPELOCK)

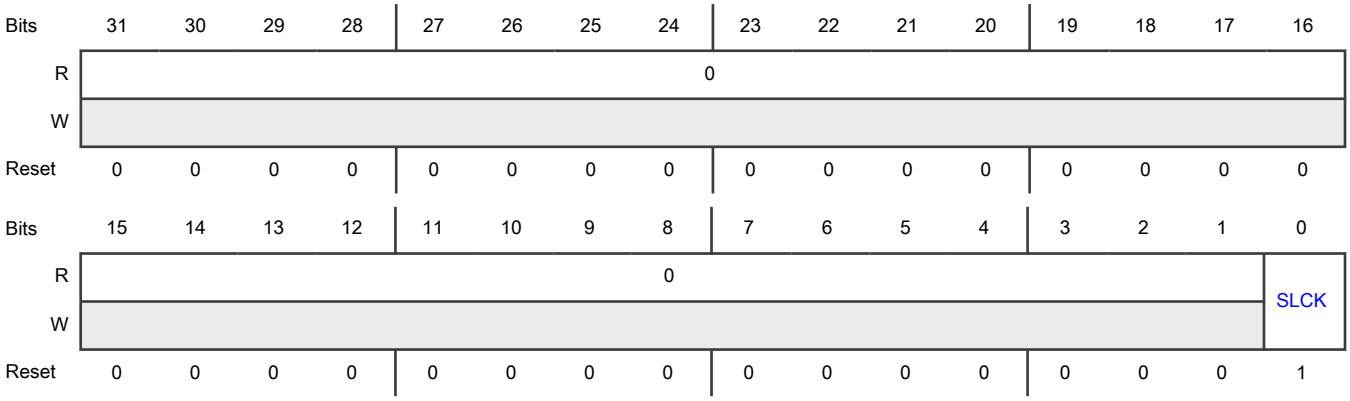
Offset

Register	Offset
PFCBLKU_SPELOCK	358h

Function

Provides a way to protect sectors from being modified. Sector protection is available on the last 256 KB of every block (for 256 KB blocks, all sectors are available for protection). Each lock bit can be associated with a particular domain ID by writing 1 to PFCBLKU_SETSLOCK[SETSLCK]. After the lock is acquired, only a master having the same domain ID can modify the corresponding lock bit. If the corresponding PFCBLKU_SETSLOCK[SETSLCK] bit is not equal to 1, any master can modify the SLCK bit. If a lock bit is already acquired by a particular domain ID, any effort to modify (1 to 0, or 0 to 1) the lock bit by a master with a different domain ID results in a transfer error.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 SLCK	Sector Lock Locks selected sector. If vector bit value = 0, the sector is available for program and erase operations. If vector bit value = 1, the sector is locked and not available for program and erase operations.

21.6.13 Block n Super Sector Program Erase Lock (PFCBLK0_SSPELOCK - PFCBLK3_SSPELOCK)

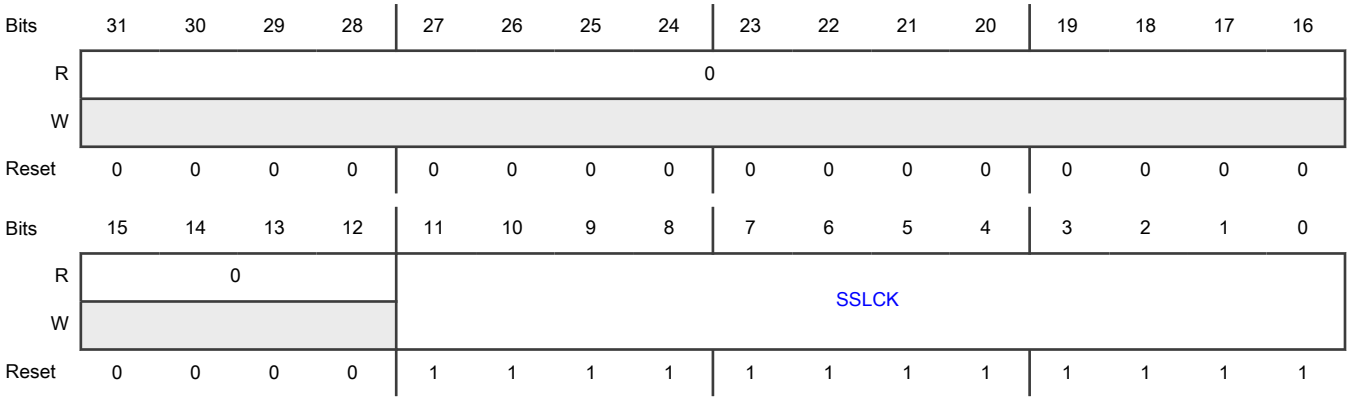
Offset

Register	Offset
PFCBLK0_SSPELOCK	35Ch
PFCBLK1_SSPELOCK	360h
PFCBLK2_SSPELOCK	364h
PFCBLK3_SSPELOCK	368h

Function

Provides a way to protect super sectors from being modified. Super sector protection is available on block space larger than 256 KB. For 256 KB blocks, this register is not applicable. For 512 KB blocks, the first half of the block is protected with super sector granularity. For 1 MB blocks, the first 768 KB is protected with super sector granularity. Each lock bit can be associated with a particular domain ID by writing 1 to the appropriate bit in PFCBLKn_SSETSLOCK[SSETSLCK]. After the lock is acquired, only a master having the same domain ID can modify the corresponding lock bit. If the corresponding PFCBLKn_SSETSLOCK[SSETSLCK] bit is not equal to 1, any master can modify the SSLCK bit. If a lock bit is already acquired by a particular domain ID, any effort to modify (1 to 0, or 0 to 1) the lock bit by a master with a different domain ID results in a transfer error.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 SSLCK	Super Sector Lock Locks selected super sector. If vector bit value = 0, the super sector is available for program and erase operations. If vector bit value = 1, the super sector is locked and not available for program and erase operations.

21.6.14 Block n Set Sector Lock (PFCBLK0_SETSLOCK - PFCBLK4_SETSLOCK)

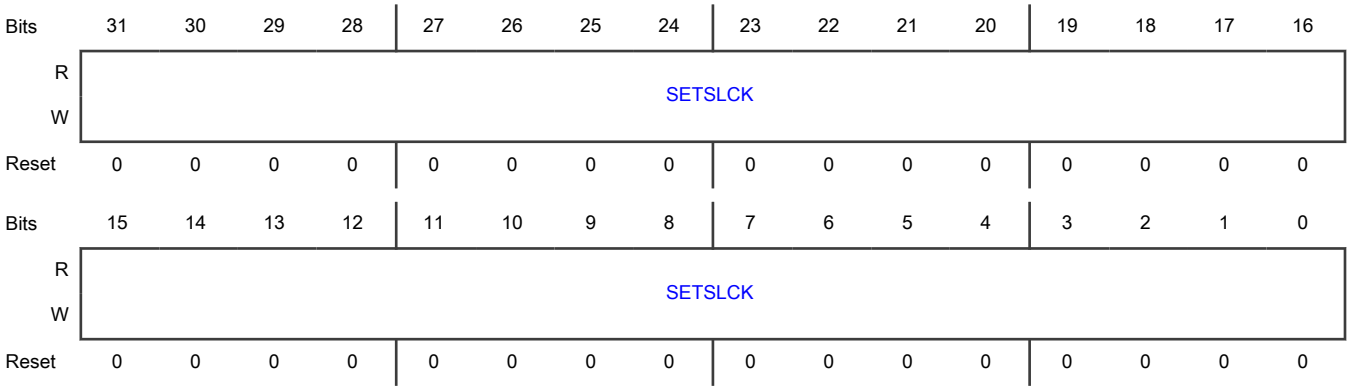
Offset

Register	Offset
PFCBLK0_SETSLOCK	380h
PFCBLK1_SETSLOCK	384h
PFCBLK2_SETSLOCK	388h
PFCBLK3_SETSLOCK	38Ch
PFCBLK4_SETSLOCK	390h

Function

Provides a mechanism for the masters to gain the ownership of the corresponding PFCBLKn_SPELOCK lock bit based on domain id . After it is equal to 1, the bit is returned to 0 at next reset. If any SETSLOCK bit is not equal to 1, the corresponding LOCK bit can be modified by any master. If a bit is already acquired by a particular domain ID, any effort to modify the lock bit by a master with a different domain ID is ignored without transfer error.

Diagram



Fields

Field	Function
31-0 SETSLCK	If the vector bit value = 0, the corresponding lock bit is not owned by any master. If the vector bit value = 1, the lock bit is owned by the masters having the domain ID stored in PFCBLKn_LOCKMASTER_Sm.

21.6.15 Block UTEST Set Sector Lock (PFCBLKU_SETSLOCK)

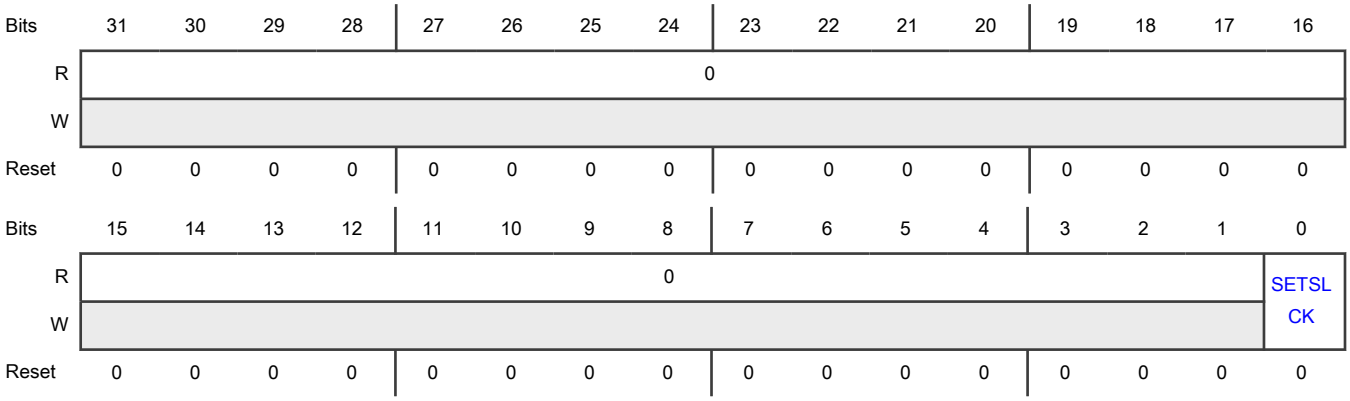
Offset

Register	Offset
PFCBLKU_SETSLOCK	398h

Function

Provides a mechanism for the masters to gain ownership of the corresponding PFCBLKU_SPELOCK lock bit based on domain id. After it is equal to 1, the bit is returned to 0 at next reset. If any SETSLOCK bit is not equal to 1, the corresponding LOCK bit can be modified by any master. If a bit is already acquired by a particular domain ID, any effort to modify the lock bit by a master with a different domain ID is ignored without transfer error.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 SETSLCK	Set Sector Lock Locks selected sector. If vector bit value = 0, the corresponding lock bit is not owned by any master. If vector bit value = 1, the lock bit is owned by the masters having the domain ID stored in PFCBLKn_LOCKMASTER_SSm.

21.6.16 Block n Set Super Sector Lock (PFCBLK0_SSETSLOCK - PFCBLK3_SSETSLOCK)

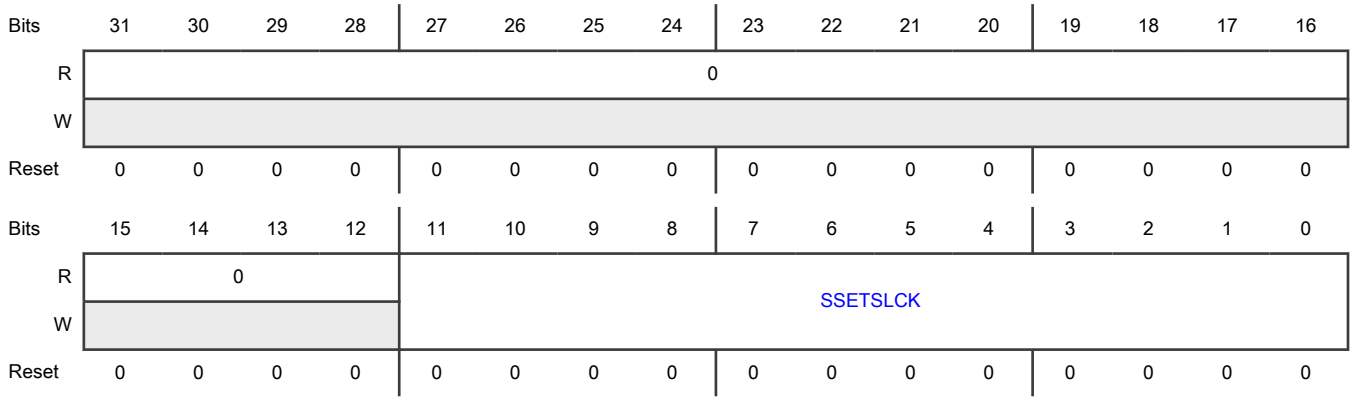
Offset

Register	Offset
PFCBLK0_SSETSLOCK	39Ch
PFCBLK1_SSETSLOCK	3A0h
PFCBLK2_SSETSLOCK	3A4h
PFCBLK3_SSETSLOCK	3A8h

Function

Provides a mechanism for the masters to gain ownership of the corresponding PFCBLKn_SPELOCK lock bit based on domain id. After it is equal to 1, the bit is returned to 0 at next reset. If any SSETSLOCK bit is not equal to 1, the corresponding LOCK bit can be modified by any master. If a bit is already acquired by a particular domain ID, any effort to modify the lock bit by a master with a different domain ID is ignored without transfer error.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 SSETSLCK	Set Super Sector Lock Locks selected super sector. If vector bit value = 0, the corresponding lock bit is not owned by any master. If vector bit value = 1, the lock bit is owned by the masters having the domain ID stored in Block a Lock Master Sector b (PFCBLK0_LOCKMASTER_S0 - PFCBLK4_LOCKMASTER_S7).

21.6.17 Block a Lock Master Sector b (PFCBLK0_LOCKMASTER_S0 - PFCBLK4_LOCKMASTER_S7)

Offset

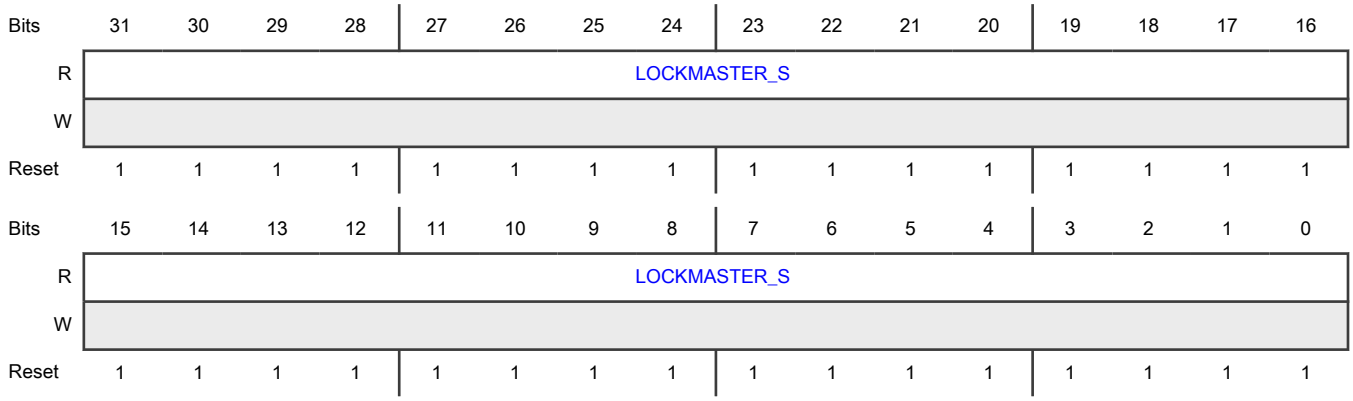
For a = 0 to 4; b = 0 to 7:

Register	Offset
PFCBLKa_LOCKMASTER_Sb	3C0h + (a × 20h) + (b × 4h)

Function

Provides the domain ID information of the master currently acquiring the lock bit. The domain ID is represented by an 8-bit field. This is a read-only register.

Diagram



Fields

Field	Function
31-0 LOCKMASTER_S	Block a Lock Master Sector b Contains domain ID of the master currently acquiring the lock bit. PFCBLK0_LOCKMASTER_S0[LOCKMASTER_S[7:0]] holds the domain ID information of PFCBLK0_SPELOCK[0]. PFCBLK0_LOCKMASTER_S0[LOCKMASTER_S[15:8]] holds the domain ID information of PFCBLK0_SPELOCK[1], and so on in incremental order.

21.6.18 Block UTEST Lock Master Sector (PFCBLKU_LOCKMASTER_S)

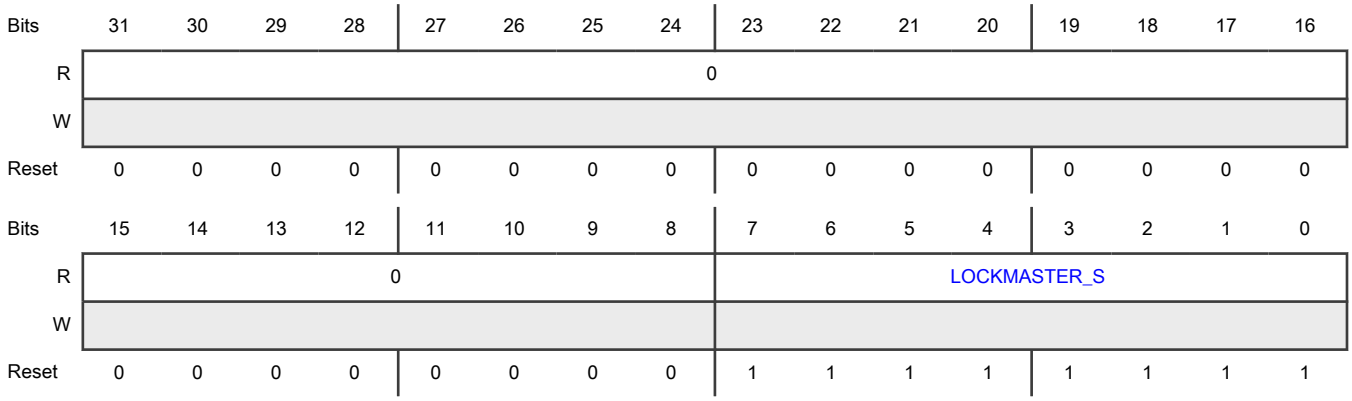
Offset

Register	Offset
PFCBLKU_LOCKMASTER_S	480h

Function

Provides the domain ID information of the master currently acquiring the lock bit. The domain ID is represented by an 8-bit field. This is a read-only register.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 LOCKMASTER_S	Lock Master Sector Contains domain ID of the master currently acquiring the lock bit. PFCBLKU_LOCKMASTER_S[LOCKMASTER_S[7:0]] holds the domain ID information of PFCBLKU_SPELOCK[0].

21.6.19 Block m Lock Master Super Sector n (PFCBLK0_LOCKMASTER_SS0 - PFCBLK3_LOCKMASTER_SS2)

Offset

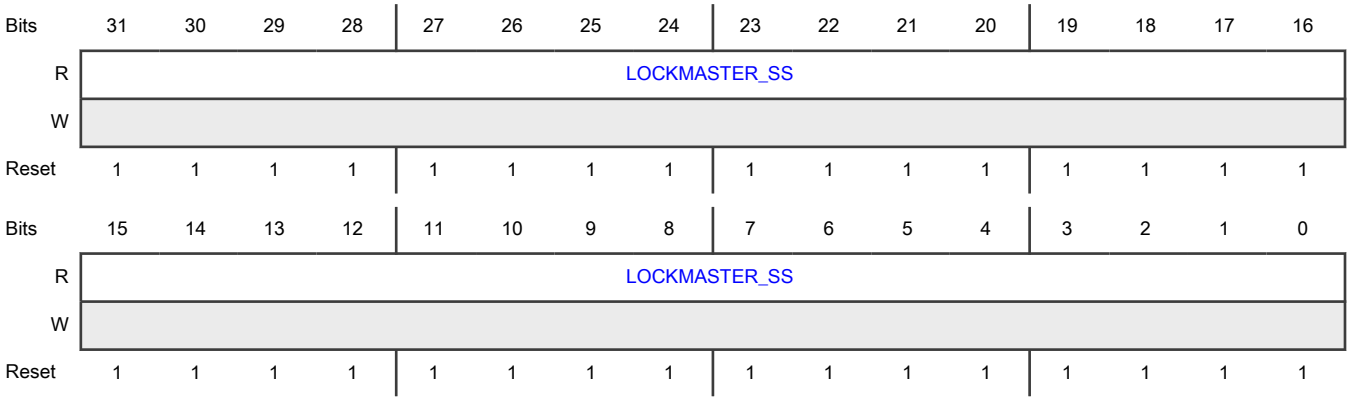
For a = 0 to 3; b = 0 to 2:

Register	Offset
PFCBLKa_LOCKMASTER_SSb	484h + (a × 10h) + (b × 4h)

Function

Provides the domain ID information of the master currently acquiring the lock bit. The domain ID is represented by an 8-bit field. This is a read-only register.

Diagram



Fields

Field	Function
31-0 LOCKMASTER_SS	Block a Lock Master Super Sector b Contains domain ID of the master currently acquiring the lock bit. PFCBLK0_LOCKMASTER_SS0[LOCKMASTER_SS[7:0]] holds thevdomain ID information of PFCBLK0_SSPELOCK[0]. PFCBLK0_LOCKMASTER_SS0[LOCKMASTER_SS[15:8]] holds the domain ID information of PFCBLK0_SSPELOCK[1], and so on in incremental order.

21.7 Glossary

AHB	Advanced high-performance bus
ECC	Error correcting code
HSE	Hardware security engine
IPS	Internal peripheral system
ID	Identification

Chapter 22
RAM Controller (PRAMC)

22.1 Chip-specific PRAMC information

22.1.1 PRAMC instances and configuration

This chip supports up to three instances of PRAMC.

Table 92. PRAMC instances and configuration

Instance	SRAM array	Applicability
PRAMC_0	SRAM 0	MWCT2015S, MWCT2016S, MWCT2D16S, MWCT2D17S
PRAMC_1	SRAM 1	MWCT2D16S, MWCT2D17S

See chapter'Memory and Memory Interfaces' for details on SRAM size.

22.1.2 RAM initialization

After chip power-on reset, you must initialize RAM to a known value using a 64-bit master before 32-bit masters can read or write to RAM. If you do not initialize RAM this way, any attempt to execute a 32-bit read or write access terminates with an uncorrectable ECC error event on this chip.

22.1.3 SRAM0 behavior while XRDC is configured to block access to SRAM0

While XRDC is configured to restrict access to SRAM0, the below behaviors should be considered:

1. In case if XRDC is configured to restrict access to SRAM0 and a read transaction to SRAM0 is performed: To ensure faster SRAM0 accesses, the access is routed to SRAM0 and blocked by PRAMC if XRDC is configured to block access. In such a case, if an error is present on the SRAM0 data, (that is error on data or error injected by EIM), the same is latched by ERM as well. This data is not used by application core since PRAMC gives a bus error in this case.
2. In case if XRDC is configured to restrict access to SRAM0 and a 32-bit write access is performed: Since SRAM is 64-bit wide, so a 32-bit write transaction is performed as a read-modify-write. In the read cycle, again as described in previous scenario, the PRAMC will initiate an error response. In this case also, if there is any error on SRAM0 data bus (i.e., error or error injected via EIM), the same is captured in ERM as well.

22.2 Introduction

The RAM controller is an interface between the system bus (AHB-Lite 2.v6) and the integrated system RAM. It converts the protocols between the system bus and the dedicated RAM array interface.

The RAM controller supports one 64-bit AHB interface and a 64-bit RAM array interface. The AHB port provides a connection to the platform crossbar for direct RAM access from the various crossbar masters. Shown below in Figure 51 is a simplified block diagram depicting the position of the RAM controller within a typical platform architecture.

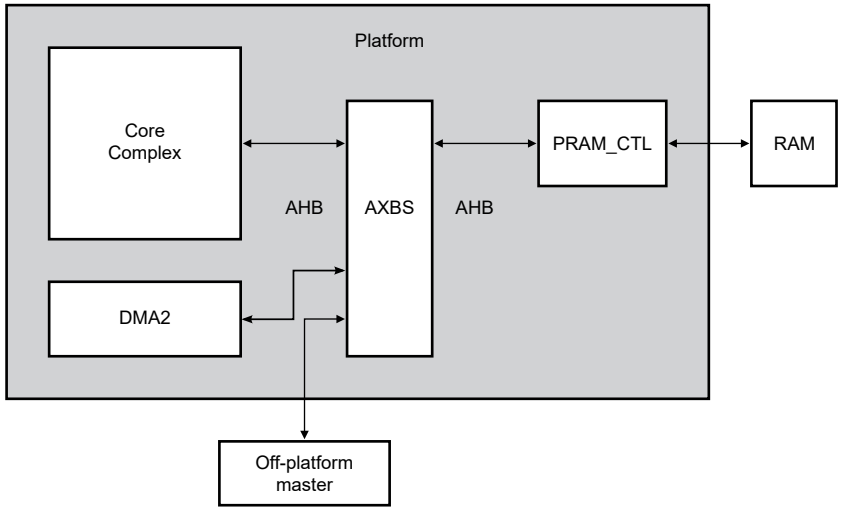


Figure 51. Simplified platform block diagram

The following list summarizes the key features of the RAM controller:

- System bus supports 64-bit data
- Array interface supports a 64-bit data + 8-bit ECC interface
- Late-write support via 64-bit data + 8-bit ECC storage buffer supports single-cycle write accesses
- Configurable read access timing (zero or one wait state programmable) allowing use in large range of frequency targets
- Read-modify-write operation supports array write size smaller than a doubleword
- Hardware EDC after ECC feature to check the ECC logic.

The address path of the RAM controller contains a MUX that chooses among the addresses presented on the system bus for a read request—either the address from the temporary holding register or the address from the late-write buffer. The temporary holding register contains the address presented during the AHB address phase of the write request. The request is delayed when being presented to the RAM by one cycle. This step makes it possible to simultaneously present the address and associated write data, which is not valid until the AHB data phase. The late-write buffer holds write requests that are delayed to facilitate single-cycle response on the system bus.

The read data path contains a MUX that chooses the source of the read data to be presented on the system bus on a read request. When a read request matches the contents of the late-write buffer, a RAM access is not required, and the late-write buffer contents are transferred to the read data bus. Otherwise, the read data is a result of a RAM access.

The write data path contains the MUX that chooses the source of the write data, as well as the control logic for generating read-modify-write operations on writes less than 64 bits in size. A write operation can be performed to empty the contents of the late-write buffer. In the case of back-to-back writes, the write may be forced to bypass the late-write buffer and instead be stored directly in RAM.

22.3 Functional description

This section describes the functions of the RAM controller.

22.3.1 Read and write operations

The RAM controller processes read and write requests to on-chip RAM and provides a register interface for access to performance tuning functions.

NOTE

The RAM controller register interface is accessible only in supervisor mode. Accesses in user mode return a transfer error.

22.3.1.1 Reads

Read transfers, of any size, can be configured to complete with a zero or one additional wait state response on the system bus.

22.3.1.2 Optional read wait state

The RAM controller can be optionally programmed to register RAM read data prior to returning it on the system bus. Setting the PRCRx[FT_DIS] field inserts a register in the read data path for use when operating the controller at high frequencies. The state of PRCRx[FT_DIS] field has no effect on writes.

A random, initial access takes two clock cycles (2T) to complete if PRCRx[FT_DIS]=0, and a WRAP4 burst will have an access time of 2-2-2-2. A random, initial access takes three clock cycles (3T) to complete if PRCRx[FT_DIS]=1, and a WRAP4 burst has an access time of 3-2-2-2.

NOTE

The number of cycles taken for a RAM access can vary ±1 clock cycle depending on the RAM speed relative to the PRAMC controller clock frequency. If system RAM is running at the same frequency as the PRAMC controller, a random initial access takes two clock cycles. If system RAM is running at a slower frequency, a random initial access may take 3 clock cycles. Subsequent burst beats take either one or two cycles depending on RAM speed relative to the PRAMC controller clock frequency.

22.3.2 Writes

This section discusses various write operations of the RAM controller.

22.3.2.1 64-bit writes

Aligned 64-bit writes execute in a single AHB data phase cycle, resulting in zero wait states on back-to-back writes of these sizes. If, during the data phase of a write, a read is requested on the AHB, the write is registered in the late-write buffer, enabling the read to take place without a wait state. The valid buffered or late-write data is stored on the next available array address phase.

Back-to-back 64-bit writes execute slightly differently. The first write bypasses the late-write buffer—the write data is stored directly to the array in the same cycle in which it is valid on the AHB.

22.3.2.2 Less than 64-bit writes

Writes of size less than 64 bits incur a read-modify-write action as a consequence of the ECC coding scheme's 64-bit granularity. In the case of a read-modify-write action, the RAM controller performs SEC/DED on the read data. The write data is merged into the appropriate byte lanes along with the potentially corrected read data. A new codeword is generated based on the newly formed doubleword. This doubleword and its associated checkbits are subsequently written to RAM. Figure 52 provides details on the read-modify-write data path.

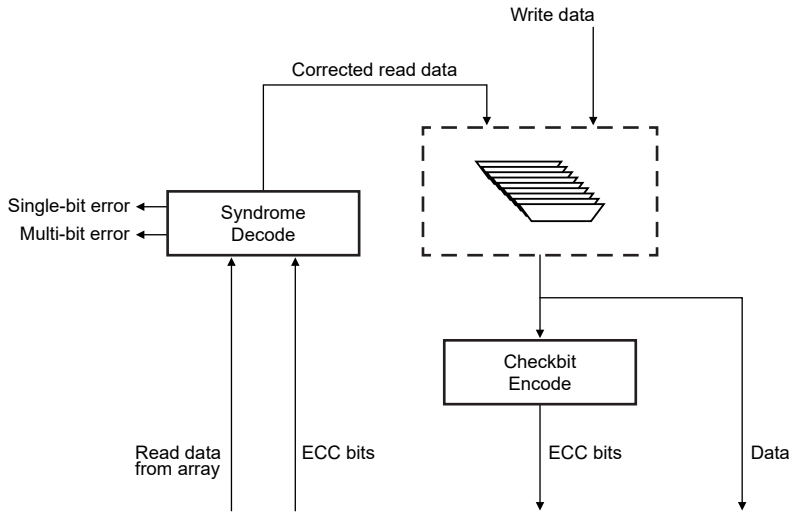


Figure 52. Read-modify-write data path

On a read-modify-write operation, the array read data is registered after it is decoded and before it is merged with the AHB write data. Therefore, writes of size less than 64 bits require the insertion of one wait state before the data phase can be completed.

22.3.2.3 Unaligned writes

The RAM controller is compliant with the AMBA-AHB2.v6 Extensions specification with regard to byte strobes. The size of the transfer is sufficient to cover all bytes being written, and covers more bytes in the case of an unaligned transfer. The address of the transfer is rounded down to the nearest boundary of the size of the transaction.

NOTE

Unaligned writes always generate read-modify-write operations in the RAM controller in order to preserve the validity of the ECC codeword.

22.4 Initialization/application information

Each memory address must be written to a known value before it is read, to initialize the ECC. This includes reads generated from the read-modify-write operation that occurs when a write transfer of less than 64 bits or an unaligned write is requested. Without first writing an address to a known value, a read from the address will most likely generate an uncorrectable ECC event.

22.5 PRAMC register descriptions

The RAM controller module provides an IPS programming model mapped to a standard on-platform peripheral slot.

NOTE

1. The programming model can only be referenced using a 32-bit (word) access. Attempted references using different access sizes or to undefined (reserved) addresses generate an IPS error termination.
2. Attempted updates to the PRAMC programming model while a PRAMC operation is in progress will result in non-deterministic behavior. Software must be architected to avoid this scenario by ensuring that PRAMC configuration changes are made only during system boot or when only one master is enabled. In multi-core devices, update the PRAMC configuration only when one core is active and no other masters, for example, DMA or communications modules, are enabled. Move any instruction execution or memory references outside the system RAM while updating the PRAMC configuration, for example, to the core local memory space.

22.5.1 PRAMC memory map

PRAMC_0 base address: 4026_4000h

PRAMC_1 base address: 4046_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Platform RAM Configuration register 1 (PRCR1)	32	RW	0000_0100h

22.5.2 Platform RAM Configuration register 1 (PRCR1)

Offset

Register	Offset
PRCR1	0h

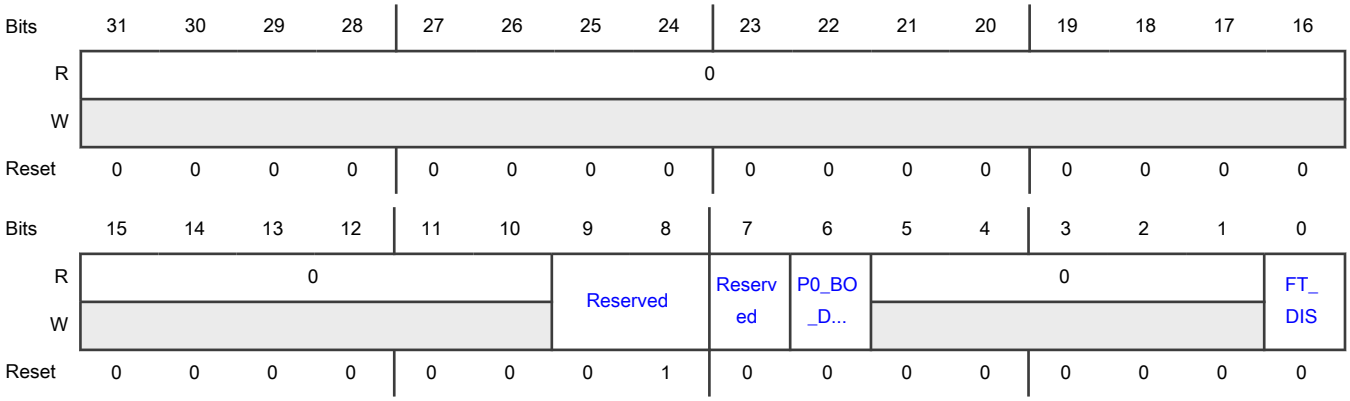
Function

The Platform RAM Configuration register 1 (PRCR1) is used to specify operation of the RAM controller.

NOTE

This register is accessible only in supervisor mode. Accesses in user mode return a transfer error.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved
6 P0_BO_DIS	Port p0 read burst optimization disable. **NOTE** The number of cycles taken for a RAM access can vary ± 1 clock cycle depending on the RAM speed relative to the PRAMC controller clock frequency. If system RAM is running at the same frequency as the PRAMC controller, a random initial access takes 2 clock cycles. If system RAM is running at a slower frequency, a random initial access may take 3 clock cycles. Subsequent burst beats take either 1 or 2 cycles depending on RAM speed relative to the PRAMC controller clock frequency. 0b - 64-bit WRP4 read bursts are optimized such that the controller returns a 2-1-1-1 response when PRCR1[FT_DIS]=1 1b - 64-bit WRP4 read bursts are not optimized; the controller returns a 2-2-2-2 response when PRCR1[FT_DIS]=1
5-1 —	Reserved
0 FT_DIS	Flow-through disabled This field defines the AHB response on reads. The state of this field has no impact on the response latency on writes. This bit is cleared by hardware reset. **NOTE** Do not change the FT_DIS bit value while accessing system RAM. Relocate code programming the FT_DIS bit to another memory area, for example, local core memory. 0b - RAM read data is passed directly to the system bus, incurring no additional latency 1b - RAM read data is registered prior to returning on the system bus, incurring one extra cycle of latency

22.6 Glossary

AHB	Arm advanced high-performance bus
DED	Double error detection
DMA	Direct memory access
MUX	Multiplexer
SEC	Single error correction

Chapter 23
Clocking

23.1 Introduction

This chapter describes the clocking architecture and includes the following information:

- System clock specifics
- Clock sources
- Clock architecture
- Clock control registers
- Clock monitoring
- Clock gating
- Module clocking

This chapter discusses the clocks generated on the chip. Peripheral-specific protocol clocks are described in the corresponding peripheral chapters.

23.2 Features

- Multiple clock sources supported for clock generation:
  - Fast internal RC oscillator (FIRC)
  - Slow internal RC oscillator (SIRC)
  - Fast external crystal oscillator (FXOSC)
  - Slow external crystal oscillator (SXOSC)

    NOTE

    SXOSC is not available in 100-MAX QFP and 48-pin LQFP packages. For details, see 'MWCT2xxxS chip's feature comparison' table in Introduction chapter.

  - Phase-locked loop (PLL)
- Frequency-modulated PLL output clock to reduce electromagnetic emissions
- Precise clocks for timers and communication functions
- Glitchless clock switching Clock Generation module (MC_CGM) clock selectors
- System clock progressive clock frequency switching (PCFS)
- Clock monitoring units (CMU_FC, CMU_FM) to check clock integrity
- Core and peripheral clock gating using the Mode Entry module's (MC_ME) partition process configuration registers

23.3 Clocking overview

The MWCT2xxxS clocking architecture consists of multiple:

- Clock sources
- Monitors
- Multiplexers
- Dividers

The blocks in the above bullet list provide the required clocking domains for the different functional blocks.

The sections in the following list show the clocking configuration of the chip:

- Clock source generation: PLL, FXOSC, FIRC, SIRC, and SXOSC
- MC_CGM mux 0 clocks: MC_CGM mux 0 generated clocks (not including EMAC clock signals)
- Clockout overview: CLKOUT_STANDBY and CLKOUT_RUN
- Other clocks
- EMAC clocking: Ethernet utilized clocking

The figures shown in MWCT2D17S and MWCT2D16S clock system diagram, MWCT2016S clock system diagram, and MWCT2015S clock system diagram show the overall clock tree for the different chip variants of the MWCT2xxxS family, which are a combination of the sections mentioned in the bullet list above.

23.3.1 Clock source generation

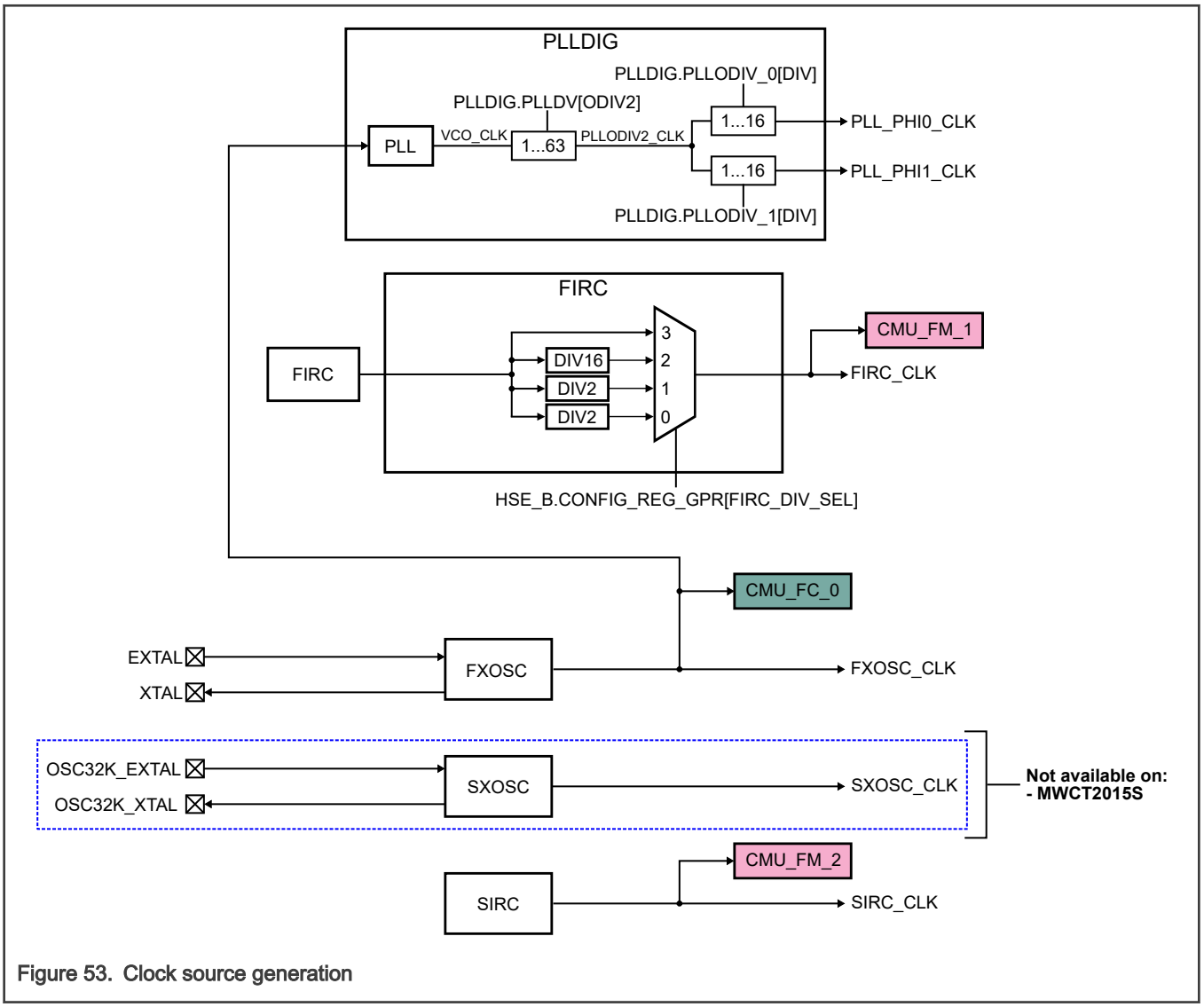


Figure 53. Clock source generation

23.3.2 MC_CGM mux 0 clocks

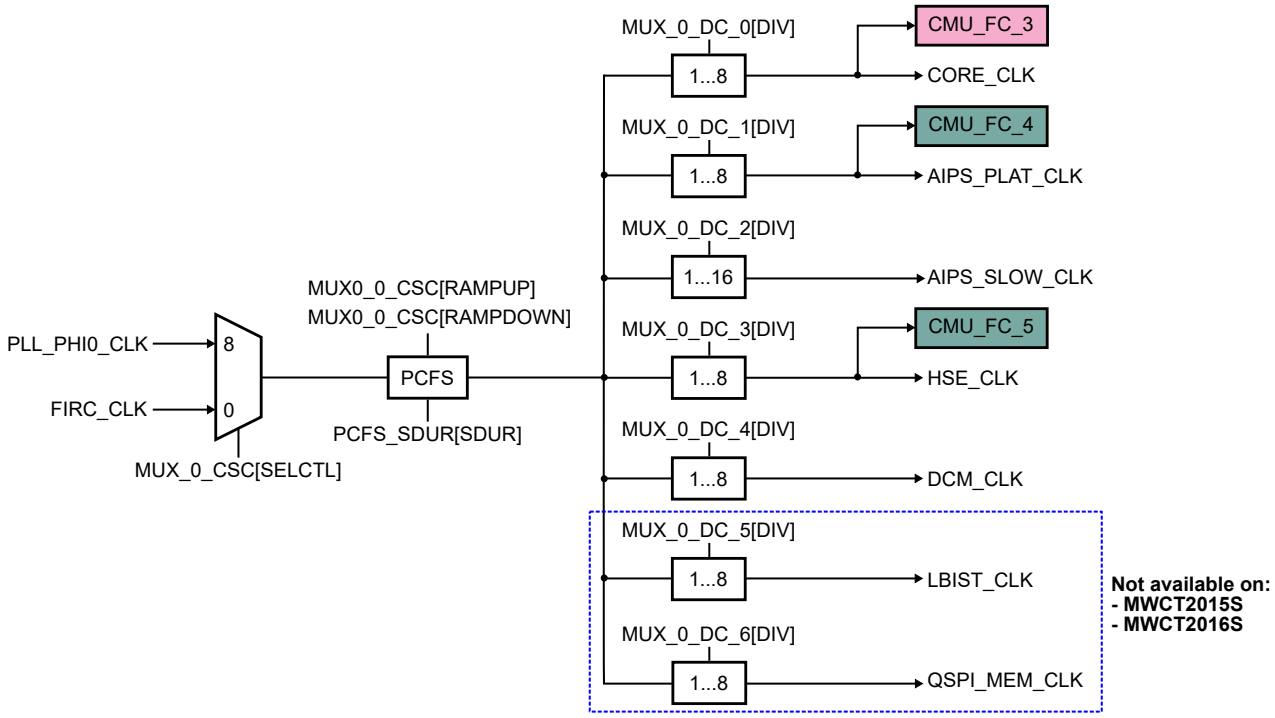


Figure 54. MC_CGM mux 0 clocks

NOTE

The clock frequency relationship between TCK and HSE_CLK clocks for HSE_B must be a minimum ratio of 1:1.5. For example, if HSE_CLK equals 80 MHz, then TCK must be less than or equal to 53 MHz (80 MHz ÷ 1.5).

23.3.3 Clockout overview

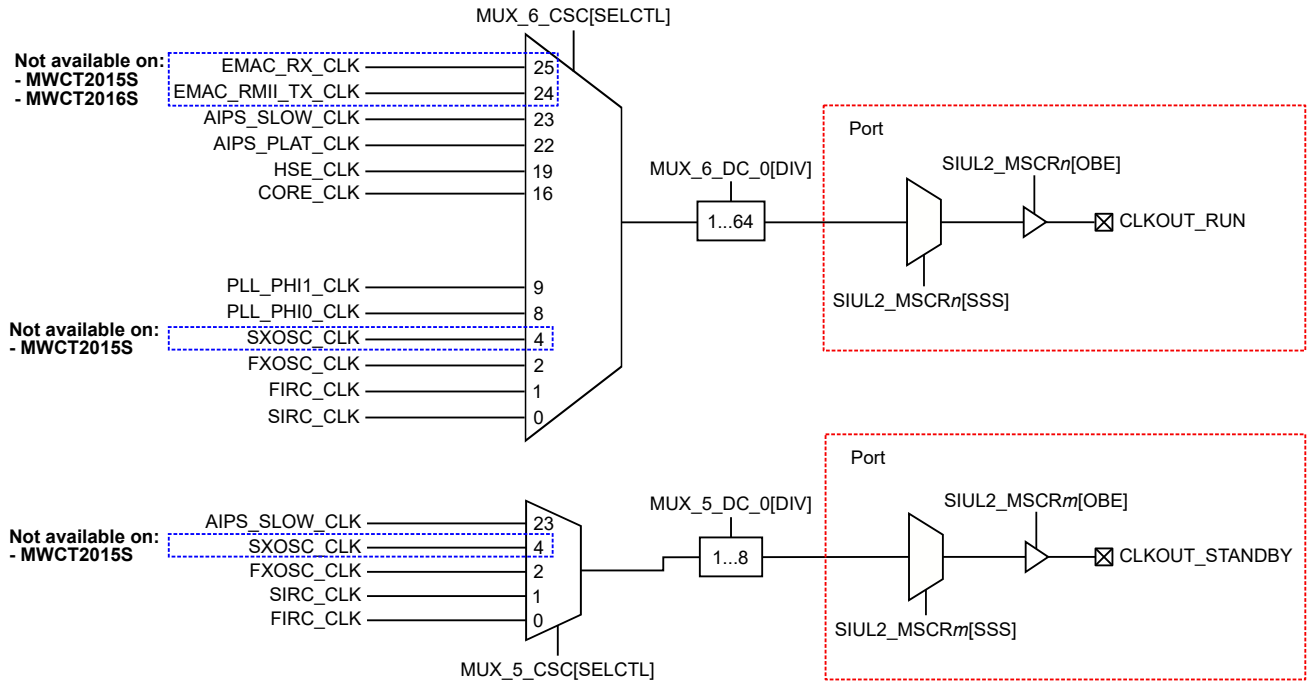


Figure 55. Clockout overview

NOTE

CLKOUT_RUN is not available during Standby mode.

23.3.3.1 SIUL2 options for CLKOUT_RUN

Table 93. SIUL2 options for CLKOUT_RUN

Source	Destination	Port	MSCR*n*	MSCR fields		
				OBE	IBE	SSS
MUX_6_DC_0 divider output	CLKOUT_RUN	PTB5	37	1	0	0101b
		PTD10	106	1	0	0110b
		PTD14	110	1	0	0111b

23.3.3.2 SIUL2 options for CLKOUT_STANDBY

Table 94. SIUL2 options for CLKOUT_STANDBY

Source	Destination	Port	MSCR*m*	MSCR fields		
				OBE	IBE	SSS
MUX_5_DC_0 divider output	CLKOUT_STANDBY	PTA12	12	1	0	0011b
		PTE10	138	1	0	0101b

23.3.4 Other clocks

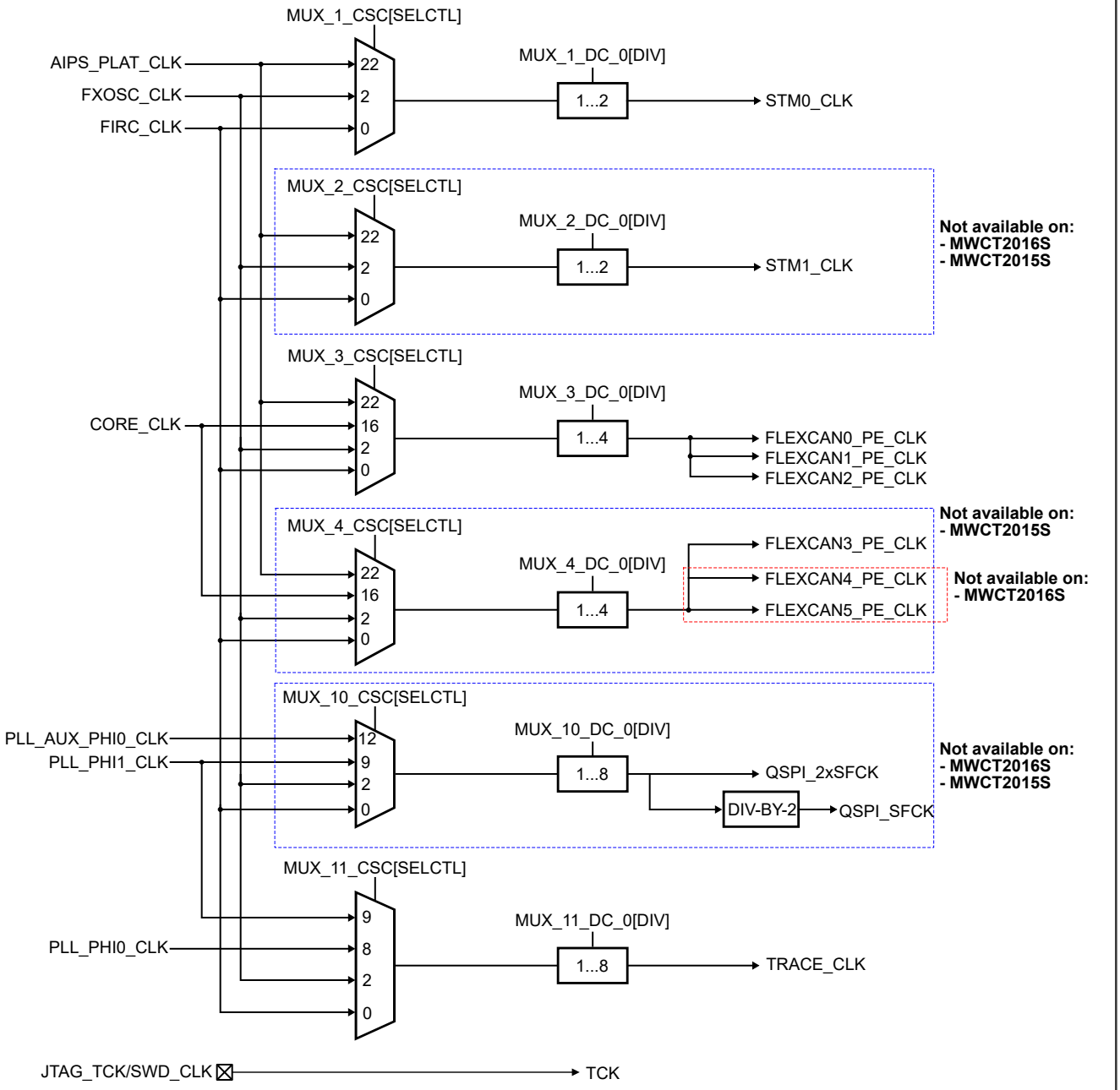


Figure 56. Other clocks

23.3.5 MWCT2D17S and MWCT2D16S clock system diagram

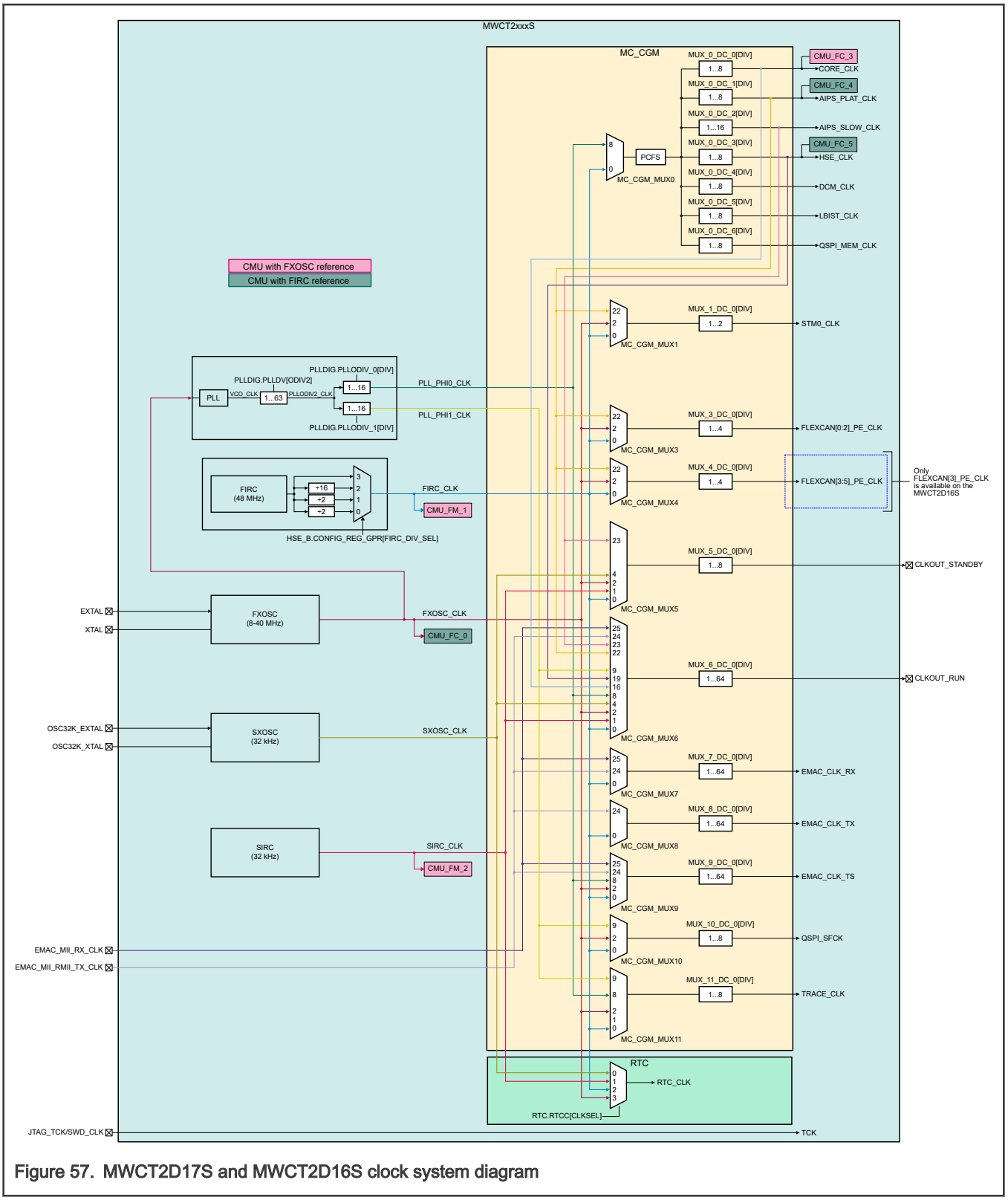


Figure 57. MWCT2D17S and MWCT2D16S clock system diagram

23.3.6 MWCT2016S clock system diagram

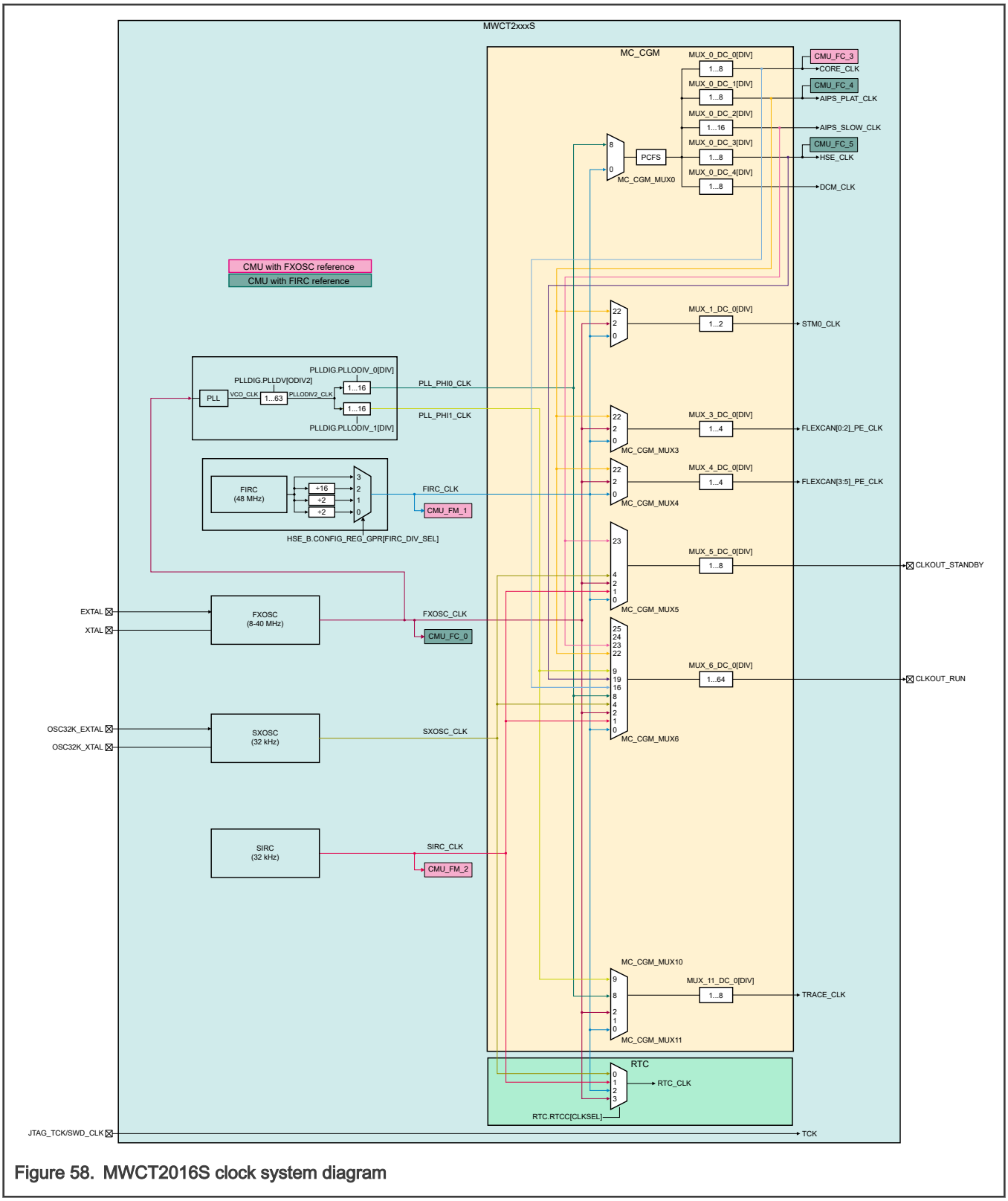


Figure 58. MWCT2016S clock system diagram

23.3.7 MWCT2015S clock system diagram

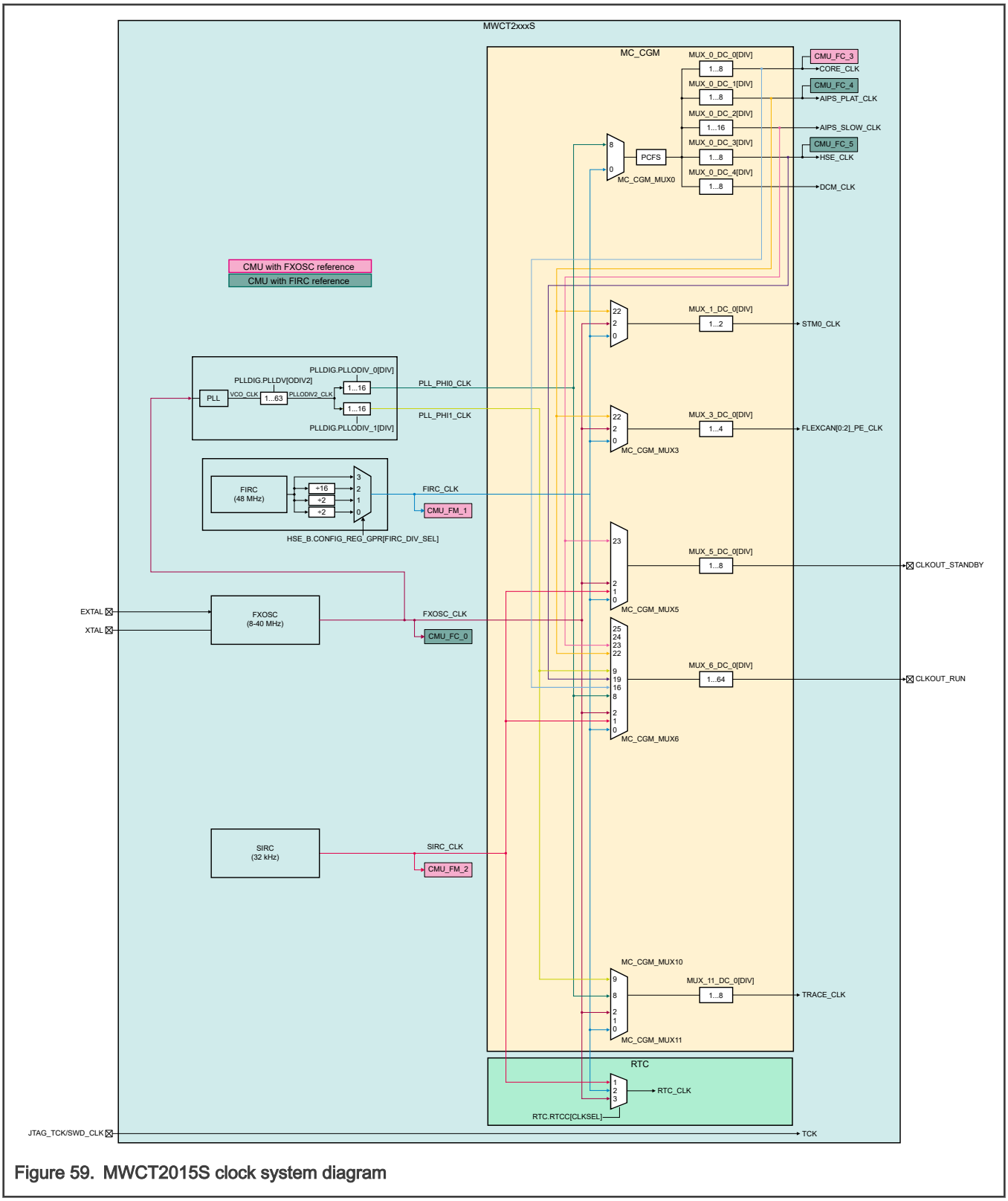


Figure 59. MWCT2015S clock system diagram

23.4 Clock sources

23.4.1 Introduction

The chip contains the following clock sourcing modules:

- FIRC
  - FIRC_CLK is the default system clock source.
- SIRC
- PLL
- FXOSC
- SXOSC (not available on the MWCT2015S)

The following list shows some of the clock system features:

- All clock sources support software configurability for enabling or disabling. [3]
- All clock sources, except SXOSC_CLK, are initialized to their default state on functional reset.
- The SXOSC_CLK supports RTC applications across functional reset and is reset on destructive reset.

Only SIRC_CLK and FIRC_CLK are enabled out of reset and are enabled on any functional reset. The other clock sources are disabled on reset.

23.4.1.1 Chip clock sources

Table 95. Chip clock sources

Clock source	Divider	Default state	Reset	Uses
FIRC_CLK (48 MHz)	1, 2, 16	On	POR (enabled on functional and destructive reset) POR assertion - FIRC_CLK disabled asynchronously POR deassertion - FIRC_CLK enabled	• Boot clock • Default system clock source • Safe clock for safety modules FCCU and FOSU • SIUL2 filter clock • MC_RGM clock source
PLL_PHI*n*_CLK (640-1280 MHz)	1...16	Off	Functional (disabled on functional reset)	• Optional system clock source • Communication modules (FlexCAN, EMAC, QuadSPI, and so on)
FXOSC_CLK (8-40 MHz)	—	Off	Functional (disabled on functional reset)	• Reference clock source for PLL • Communication modules (FlexCAN, EMAC, QuadSPI, and so on)
SXOSC_CLK (32.768 KHz)	—	Off	Destructive (disabled on destructive reset)	• RTC source for operation across functional reset (SXOSC_CLK is not available on the MWCT2015S)

Table continues on the next page...

[3] FIRC_CLK and SIRC_CLK cannot be disabled during Run mode.

Table 95. Chip clock sources (continued)

Clock source	Divider	Default state	Reset	Uses
SIRC_CLK (32 KHz)	—	On	POR (enabled on functional reset)	• Safe clock along with FIRC_CLK • SWT clock source • POR_WDG source clock

23.4.2 Chip input clocks

Table 96. Chip input clocks

Pin	Description
XTAL	FXOSC crystal pins
EXTAL	
OSC32K_XTAL	SXOSC crystal pins (not available on the MWCT2015S)
OSC32K_EXTAL	
EMAC_MII_RMII_TX_CLK	EMAC transmitter clock/EMAC RMII clock (not available on the MWCT2015S or MWCT2016S)
EMAC_MII_RX_CLK	EMAC receiver clock (not available on the MWCT2015S or MWCT2016S)
JTAG_TCLK/SWD_CLK	JTAG/SWD clock
SAI*n*_MCLK	SAI_*n* clock in slave mode (not available on the MWCT2015S or MWCT2016S)
SAI*n*_BCLK	SAI_*n* bit clock in slave mode (not available on the MWCT2015S or MWCT2016S)
LPSPI*n*_SCK	LPSPI*n* serial clock in slave mode
LPI2C*n*_SCL	LPI2C_*n* clock
LPI2C*n*_SCLS	LPI2C_*n* secondary clock

23.4.3 Chip output clocks

Table 97. Chip output clocks

Pin	Description
CLKOUT_RUN	Available during Run mode, unavailable during Standby mode
CLKOUT_STANDBY	Available during both Run and Standby modes
LPSPI*n*_SCK	LPSPI*n* serial clock in master mode
LPI2C*n*_SCL	LPI2C_*n* clock
LPI2C*n*_SCLS	LPI2C_*n* secondary clock
EMAC_MII_RMII_MDC	EMAC clock for control data transfer to PHY (not available on MWCT2015S or MWCT2016S)
EMAC_MII_RMII_TX_CLK	EMAC transmit clock (not available on MWCT2015S or MWCT2016S)

Table continues on the next page...

Table 97. Chip output clocks (continued)

Pin	Description
TRACE_ETM_CLKOUT	ETM trace clock (not available on MWCT2015S, MWCT2016S, or MWCT2D16S)
SAI*n*_BCLK	SAI_*n* bit clock in master mode (not available on MWCT2015S or MWCT2016S)
QuadSPI_SCKFA	QuadSPI serial clock for serial flash device A (fast) (not available on MWCT2015S or MWCT2016S)

23.4.4 Fast internal RC oscillator (FIRC)

The chip has an FIRC with the following features:

- Acts as the system clock source on power-up and after any reset event.
  - Important to detect FIRC_CLK failure and recovery
- Acts as the chip's safe clock for safety-relevant applications.
- Is always enabled in Run mode and can be optionally enabled in Standby mode.
- Used as clock source for the following:
  - MC_RGM
  - FCCU and FOSU
  - SIUL2 filters

23.4.4.1 FIRC failure detection

The FIRC_CLK is the safe clock source used as the FCCU and FOSU clock source. The chip supports FIRC_CLK failure detection and recovery by the mechanisms described in the following cases:

- Case 1 - FIRC_CLK not used as system clock and goes out of range:
  - CMU_FM_1 continuously measures the FIRC_CLK clock frequency using FXOSC_CLK as the reference. On each metering window completion, CMU_FM_1 asserts an interrupt (if configured, CMU_FM_1.IER[FMCIE] is 1). You store a configured reference clock count CMU_FM_1.RCCR[REF_CNT] (for example, number of FXOSC_CLK cycles in metering window). Your software checks the FIRC_CLK clock counts by reading CMU_FM_1.SR[MET_CNT].
  - In the event FIRC_CLK goes out of range, the software detects the frequency variation after the subsequent metering window by checking CMU_FM_1.SR[MET_CNT] and takes necessary action by either of the recommended options:
    - SBC-driven power cycle: The SBC receives indication from the chip (through GPIO toggle, QuadSPI communication, and so on). The SBC initiates a power cycle sequence.
    - Software-driven functional reset: The chip executes a functional reset as configured by your software.
- Case 2 - FIRC_CLK not used as system clock and fails (becomes stuck):
  - CMU_FM_1 continuously measures the FIRC_CLK frequency with FXOSC_CLK as the reference. Application software must check FIRC_CLK after a defined time limit by reading CMU_FM_1.SR[MET_CNT] and CMU_FM_1.SR[FMTO].
  - If there is an FIRC_CLK failure, the CMU_FM_1 writes a 1 to the timeout status flag CMU_FM_1.SR[FMTO].
  - When CMU_FM_1.SR[FMTO] is 1, application software takes necessary action by an SBC-driven power cycle wherein the chip provides an indication to the SBC (through GPIO toggle, QuadSPI communication, and so on). The SBC then initiates a power cycle sequence.
- Case 3 - FIRC_CLK used as system clock and goes out of range or fails (becomes stuck):
  - CMU_FC_3, CMU_FC_4, and CMU_FC_5 continuously monitor the system clock nodes with FXOSC_CLK as reference for FLL or FHH events.

- In the event of FIRC_CLK failure (when FIRC_CLK acts as the system clock source), these CMUs report the FLL event, acting as a destructive reset source.
- The system then undergoes the reset sequence, during which the FIRC_CLK is reinitialized.
- The MC_RGM.DES fields indicate the source of the reset event.
- In addition, CMU_FM_1 monitors FIRC_CLK. If the software is not able to service the CMU_FM_1 interrupt before POR_WDG timeout, POR_WDG treats this as a critical FIRC_CLK failure and initiates a POR_WDG recovery. Therefore, you must ensure that, if enabled, the CMU_FM_1 interrupt must be serviced within the POR_WDG timeout duration.

23.4.4.2 FIRC_CLK behavior in Standby mode

FIRC_CLK can be optionally enabled in Standby mode by configuring FIRC.STDBY_ENABLE[STDBY_EN].

When the PMC acknowledges the Standby mode entry, the FIRC_CLK switches from the On state to the Standby mode configuration selected by the FIRC.STDBY_ENABLE[STDBY_EN] configuration. On wakeup from Standby mode, the FIRC_CLK configuration switches from the Standby mode configured state to the On state.

The FIRC at 3 MHz is meant to be used with the Low Power Run Mode only. If the chip needs to enter Standby mode, then the FIRC must be configured to 48 MHz with the FIRC_DIV_SEL as '0b11' during the Standby Mode. After exiting the Standby mode, the FIRC can be changed to 3 MHz if desired.

NOTE

When the trims are being applied, the FIRC will appear as momentarily disabled as shown in Reset timing diagram in the Reset overview chapter.

23.4.5 SIRC

The chip has a SIRC having the following features:

- Is always enabled in Run mode and can be optionally enabled in Standby mode. Having the SIRC always enabled improves system robustness by ensuring that a clock is always available for various SWTs when reducing the chip power consumption in Standby mode.
- Used as clock source for the following:
  - SWT
  - POR_WDG

23.4.5.1 SIRC failure detection

Like the FIRC_CLK, the SIRC_CLK is a safe clock for the design. The SIRC_CLK is used as clock source for SWTs and POR_WDG and therefore it is important to detect SIRC failure and ensure its recovery. The chip supports SIRC_CLK failure detection and recovery by the mechanism described below.

- Case 1 - SIRC_CLK goes out of range:
  - CMU_FM_2 continuously measures the SIRC_CLK clock frequency with FXOSC_CLK as reference. On each metering window completion, the CMU_FM_2 raises an interrupt. The software checks the SIRC clock counts by CMU_FM_2.SR[MET_CNT] with respect to the reference clock counts CMU_FM_2.RCCR[REF_CNT].
  - In the event of clock going out of range, the software detects the frequency variation after the subsequent metering window by checking CMU_FM_2.SR[MET_CNT] and takes necessary action by either of the recommended options:
    1. SBC-driven power cycle. The chip gives an indication to the SBC (through GPIO toggle, QuadSPI communication, and so on.). The SBC then initiates a power cycle sequence.
    2. SW-driven functional reset. The chip executes a functional reset by software.
- Case 2 - SIRC_CLK fails (becomes stuck):

- CMU_FM_2 continuously measures the SIRC clock frequency with FXOSC_CLK as reference. Software needs to check this reference after a predefined time by checking CMU_FM_2.SR[MET_CNT] and CMU_FM_2.SR[FMTO].
- In the event of SIRC_CLK clock failure, the CMU_FM_2 writes 1 to the timeout status flag field in its status register, namely, CMU_FM_2.SR[FMTO].
- When CMU_FM_2.SR[FMTO] is 1, the software takes necessary action by either of the recommended options:
    1. SBC-driven power cycle: The chip gives an indication to the SBC (through GPIO toggle, QuadSPI communication, and so on). The SBC then initiates a power cycle sequence.
    2. Software-driven functional reset: User software executes a functional reset.

23.4.5.2 SIRC behavior in Standby mode

SIRC can optionally be enabled in Standby mode by configuring SIRC.MISCELLANEOUS_IN[STANDBY_ENABLE].

When PMC acknowledges the Standby mode entry, the SIRC switches from the On state to the standby configuration selected by SIRC.MISCELLANEOUS_IN[STANDBY_ENABLE] configuration. On wake-up from Standby mode, the SIRC configuration switches back from the standby-configured state to the On state.

NOTE

When the trims are being applied, the SIRC will appear as momentarily disabled similar to FIRC, as shown in section Reset timing diagram.

23.4.6 FXOSC

The chip supports an 8–40 MHz fast crystal oscillator which has following features:

- Acts as the reference for PLL.
- Supports crystal input mode and bypass mode if using an external oscillator.
- Acts as a clock source for communication modules:
    - FlexCAN
    - QuadSPI[4]
    - EMAC (EMAC_CLK_TS)[4]

23.4.7 SXOSC

The chip supports a slow crystal oscillator (SXOSC) which has the following features (not available on the MWCT2015S):

- Supports crystal input mode.
- Acts as a clock source for RTC. As SXOSC is not affected by functional reset, it supports RTC operation across functional reset. SXOSC is only reset on destructive reset.

23.4.8 PLL

The chip contains up to two PLL to provide precision clock source with the following features:

- Optional system clock source (in high performance applications)
- System clock source in safety applications
- Can be used as clock source for communication modules, when configured as system clock source:
    - FlexCAN (in SYNC mode operation)
    - EMAC (not available on MWCT2015S or MWCT2016S)

[4] Not present in MWCT2015S & MWCT2016S.

  - QuadSPI (not available on MWCT2015S or MWCT2016S)
  - LPSPI
  - LPI2C
  - FlexIO
  - LPUART
- Supports frequency modulation
- Contains lock status monitoring logic which supports loss-of-lock indication

In case of a loss of lock after lock has been acquired, the Second PLL will generate a loss of lock flag. At Power-Up, the Second PLL shall be disabled. It will be up to the application software to enable it when needed. The default reaction of loss of lock flag from the Second PLL will be an interrupt. The reaction of a loss of lock from the Second PLL will be optionally configured as a functional reset, controlled by GPR.

The Second PLL supports a clock output frequency of up to 250MHz (nominal), for the FXOSC input frequencies.

NOTE

See PLL Digital Interface (PLLDIG) for PLL configuration details. See the *MWCT2xxxS Data Sheet* for PLL limitations.

23.4.8.1 PLL configurations

The PLL output predivider frequency depends on the PLLDIG.PLLDV[RDIV] and PLLDIG.PLLDV[MFI] configurations. The PLL VCO clock can be divided further by configuring PLLDIG.PLLODIV_0[DIV] for PLL_PHI0_CLK and PLLDIG_PLLODIV_1[DIV] for PLL_PHI1_CLK (see the PLL Digital Interface (PLLDIG) chapter for configuration details).

23.4.8.1.1 PLL configuration sequence

Before enabling the PLL, you must enable FXOSC_CLK and wait until it is stable. FXOSC.STAT[OSC_STAT] must be monitored to determine the FXOSC_CLK status.

To disable the PLL, the software must disable PLL first and only then disable FXOSC (if required).

23.4.9 Chip clock outputs

The chip supports two CLKOUT_*x* pins for allowing viewing of some internal clocks as follows:

- CLKOUT_STANDBY
  - Used for showing clocks available in Run and Standby modes.
- CLKOUT_RUN
  - Used for showing only Run mode clocks.

See the Clockout overview section and the Clock Generation Module (MC_CGM) chapter for details on available clocks and configuration.

NOTE

The CLKOUT_STANDBY registers are latched when the chip enters Standby mode and are reset in Standby mode sequence. Therefore, the CLKOUT_STANDBY signal needs to be reconfigured on Standby mode exit.

NOTE

CLKOUT_STANDBY is available on two pads GPIO[12] and GPIO[138] but CLKOUT across functional reset and standby is supported only on GPIO[12] and OBE(output buffer enable) is controlled by DCM GPR bit. Please refer to DCMRWP1[3] bit for detail.

23.5 MC_CGM

The MC_CGM controls the clock functionality of the chip. See the Clock Generation Module (MC_CGM) chapter for details on MC_CGM clocking controls.

23.5.1 MC_CGM clock multiplexer types

In this chip, CLKOUT_RUN, CLKOUT_STANDBY, and TRACE_CLK multiplexers are software-controlled multiplexers. The rest are hardware-controlled multiplexers (see the Clock Generation Module (MC_CGM) chapter for details on software and hardware multiplexers).

23.5.2 MC_CGM clock multiplexers for MWCT2015S and MWCT2016S

Table 98. MC_CGM clock multiplexers for MWCT2015S and MWCT2016S

Clock mux	Register description	Source inputs [1]	Register	Selector output	Divider output
Clock mux 0	Select Control	FIRC_CLK PLL_PHI0_CLK	MUX_0_CSC		—
	Select Status		MUX_0_CSS		
	Divider 0 Control	—	MUX_0_DC_0		CORE_CLK
	Divider 1 Control		MUX_0_DC_1	—	AIPS_PLAT_CLK
	Divider 2 Control		MUX_0_DC_2		AIPS_SLOW_CLK
	Divider 3 Control		MUX_0_DC_3		HSE_CLK
	Divider 4 Control		MUX_0_DC_4		DCM_CLK
Clock mux 3	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK	MUX_3_CSC		—
	Select Status		MUX_3_CSS		
	Divider 0 Control	—	MUX_3_DC_0	—	FLEXCAN0_PE_CLK FLEXCAN1_PE_CLK FLEXCAN2_PE_CLK
Clock mux 4 [2]	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK	MUX_4_CSC		—
	Select Status		MUX_4_CSS		
	Divider 0 Control	—	MUX_4_DC_0	—	FLEXCAN3_PE_CLK FLEXCAN4_PE_CLK FLEXCAN5_PE_CLK
Clock mux 5 [3]	Select Control	FIRC_CLK SIRC_CLK FXOSC_CLK SXOSC_CLK AIPS_SLOW_CLK	MUX_5_CSC		—
	Select Status		MUX_5_CSS	—	
	Divider 0 Control	—	MUX_5_DC_0		CLKOUT_STANDBY

Table continues on the next page...

Table 98. MC_CGM clock multiplexers for MWCT2015S and MWCT2016S (continued)

Clock mux	Register description	Source inputs [1]	Register	Selector output	Divider output
Clock mux 6 [4]	Select Control	FIRC_CLK SIRC_CLK FXOSC_CLK SXOSC_CLK PLL_PHI0_CLK PLL_PHI1_CLK CORE_CLK HSE_CLK AIPS_PLAT_CLK AIPS_SLOW_CLK EMAC_RMII_TX_CLK EMAC_RX_CLK	MUX_6_CSC	—	—
	Select Status		MUX_6_CSS		
	Divider 0 Control	—	MUX_6_DC_0		CLKOUT_RUN
Clock mux 11	Select Control	FIRC_CLK FXOSC_CLK PLL_PHI0_CLK PLL_PHI1_CLK	MUX_11_CSC	—	—
	Select Status		MUX_11_CSS		
	Divider 0 Control	—	MUX_11_DC_0		TRACE_CLK

1. The default clock selected for all clock mux selectors is FIRC_CLK (out of reset).
2. Clock mux 4 and FLEXCAN[3:5]_PE_CLK are not available on the MWCT2015S.
3. SXOSC as source for clock mux 5 is not available on the MWCT2015S.
4. SXOSC as source for clock mux 6 is not available on the MWCT2015S.

23.5.3 MC_CGM clock multiplexers (excluding MWCT2015S and MWCT2016S)

Table 99. MC_CGM clock multiplexers (excluding MWCT2015S and MWCT2016S)

Clock mux	Register description	Source inputs [1]	Register	Selector output	Divider output
Clock mux 0	Select Control	FIRC_CLK PLL_PHI0_CLK	MUX_0_CSC	—	—
	Select Status		MUX_0_CSS		
	Divider 0 Control	—	MUX_0_DC_0		CORE_CLK
	Divider 1 Control		MUX_0_DC_1		AIPS_PLAT_CLK
	Divider 2 Control		MUX_0_DC_2		AIPS_SLOW_CLK
	Divider 3 Control		MUX_0_DC_3		HSE_CLK
	Divider 4 Control		MUX_0_DC_4		DCM_CLK
	Divider 5 Control		MUX_0_DC_5		LBIST_CLK
	Divider 6 Control		MUX_0_DC_6		QSPI_MEM_CLK

Table continues on the next page...

Table 99. MC_CGM clock multiplexers (excluding MWCT2015S and MWCT2016S) (continued)

Clock mux	Register description	Source inputs [1]	Register	Selector output	Divider output
Clock mux 1	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK	MUX_1_CSC	—	—
	Select Status		MUX_1_CSS		
	Divider 0 Control	—	MUX_1_DC_0		STM0_CLK
Clock mux 2	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK	MUX_2_CSC	—	—
	Select Status		MUX_2_CSS		
	Divider 0 Control	—	MUX_2_DC_0		STM1_CLK
Clock mux 3	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK	MUX_3_CSC	—	—
	Select Status		MUX_3_CSS		
	Divider 0 Control	—	MUX_3_DC_0		FLEXCAN0_PE_CLK FLEXCAN1_PE_CLK FLEXCAN2_PE_CLK
Clock mux 4	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK	MUX_4_CSC	—	—
	Select Status		MUX_4_CSS		
	Divider 0 Control	—	MUX_4_DC_0		FLEXCAN3_PE_CLK FLEXCAN4_PE_CLK[2] FLEXCAN5_PE_CLK[2]
Clock mux 5	Select Control	FIRC_CLK SIRC_CLK FXOSC_CLK SXOSC_CLK AIPS_SLOW_CLK	MUX_5_CSC	—	—
	Select Status		MUX_5_CSS		
	Divider 0 Control	—	MUX_5_DC_0		CLKOUT_STANDBY
Clock mux 6	Select Control	FIRC_CLK SIRC_CLK FXOSC_CLK SXOSC_CLK PLL_PHI0_CLK PLL_PHI1_CLK CORE_CLK HSE_CLK AIPS_PLAT_CLK AIPS_SLOW_CLK EMAC_RMII_TX_CLK EMAC_RX_CLK	MUX_6_CSC	—	—
	Select Status		MUX_6_CSS		

Table continues on the next page...

Table 99. MC_CGM clock multiplexers (excluding MWCT2015S and MWCT2016S) (continued)

Clock mux	Register description	Source inputs [1]	Register	Selector output	Divider output
	Divider 0 Control	—	MUX_6_DC_0		CLKOUT_RUN
Clock mux 7	Select Control	FIRC_CLK EMAC_RMII_MII_TX_CLK (pin) EMAC_RX_CLK (pin)	MUX_7_CSC	—	—
	Select Status		MUX_7_CSS		
	Divider 0 Control	—	MUX_7_DC_0		EMAC_RX_CLK
Clock mux 8	Select Control	FIRC_CLK EMAC_RMII_MII_TX_CLK (pin)	MUX_8_CSC	—	—
	Select Status		MUX_8_CSS		
	Divider 0 Control	—	MUX_8_DC_0		EMAC_TX_CLK
Clock mux 9	Select Control	FIRC_CLK EMAC_RMII_MII_TX_CLK (pin) EMAC_RX_CLK (pin)	MUX_9_CSC	—	—
	Select Status		MUX_9_CSS		
	Divider 0 Control	—	MUX_9_DC_0		EMAC_TS_CLK
Clock mux 10	Select Control	FIRC_CLK FXOSC_CLK PLL_PHI0_CLK PLL_PHI1_CLK EMAC_RMII_MII_TX_CLK (pin) EMAC_RX_CLK (pin)	MUX_10_CSC	—	—
	Select Status		MUX_10_CSS		
	Divider 0 Control	—	MUX_10_DC_0		QSPI_SFCK
Clock mux 11	Select Control	FIRC_CLK FXOSC_CLK PLL_PHI0_CLK PLL_PHI1_CLK	MUX_11_CSC	—	—
	Select Status		MUX_11_CSS		
	Divider 0 Control	—	MUX_11_DC_0		TRACE_CLK

1. The default clock selected for all clock mux selectors is FIRC_CLK (out of reset).
2. FLEXCAN[4:5]_PE_CLK are not available on the MWCT2D16S.

23.5.4 MC_CGM clock sources mapping

Table 100. MC_CGM clock sources mapping

Clock selector index [1]	MC_CGM clock source	Clock source
0	clk_src_0	FIRC_CLK
1	clk_src_1	SIRC_CLK

Table continues on the next page...

Table 100. MC_CGM clock sources mapping (continued)

Clock selector index [1]	MC_CGM clock source	Clock source
2	clk_src_2	FXOSC_CLK
3	Reserved	Reserved
4	clk_src_4	SXOSC_CLK [2]
5–7	Reserved	Reserved
8	clk_src_8	PLL_PHI0_CLK
9	clk_src_9	PLL_PHI1_CLK
10–14	Reserved	Reserved
15	Reserved	Reserved
16	clk_src_16	CORE_CLK
17–18	Reserved	Reserved
19	clk_src_19	HSE_CLK
20–21	Reserved	Reserved
22	clk_src_22	AIPS_PLAT_CLK
23	clk_src_23	AIPS_SLOW_CLK
24	clk_src_24	EMAC_RMII_TX_CLK [3]
25	clk_src_25	EMAC_RX_CLK [3]
26–49	Reserved	Reserved
50	clk_src_50	CLKOUT_RUN

1. All clock selector indexes not shown are reserved.
2. SXOSC_CLK is not available on the MWCT2015S.
3. EMAC_RMII_TX_CLK and EMAC_RX_CLK are not available on the MWCT2015S or MWCT2016S.

23.6 Peripheral clocking

The module clocking diagrams for the peripherals are shown in the following subsections (see Peripheral clock gating for peripheral clock gating possibilities).

23.6.1 Module clocking

The following sections show how the chip modules use MODULE_CLK and REG_INTF_CLK to control their functionality.

23.6.1.1 Communication modules

Figure 60 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 101 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

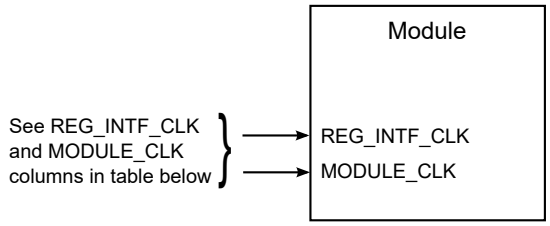


Figure 60. Communication module clocks

Table 101. Communication module clocking

Module	MODULE_CLK	REG_INTF_CLK
LPSPI	See LPSPI*n* clocking.	
LPI^{2}C	See LPI2C_*n* clocking.	
FlexIO	See FlexIO clocking.	
FlexCAN	See FlexCAN_*n* clocking.	
SAI	See SAI_*n* clocking.	
EMAC	See EMAC clocking.	
LPUART_[0,8]	AIPS_PLAT_CLK	AIPS_PLAT_CLK
LPUART_[1:7,9:15]	AIPS_SLOW_CLK	AIPS_SLOW_CLK
QuadSPI	See QuadSPI clocking.	

23.6.1.1.1 FlexCAN_*n* clocking

The figure below shows the FlexCAN_*n* clocking.

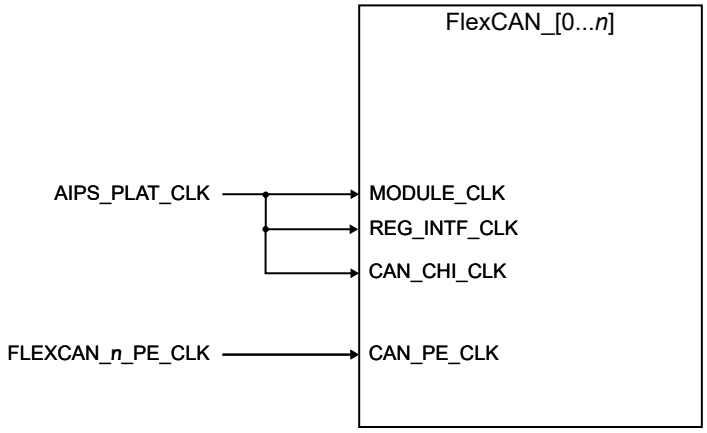


Figure 61. FlexCAN_*n* clocking

FlexCAN has the following unique clocks:

- CAN_CHI_CLK—FlexCAN controller host interface clock
- CAN_PE_CLK—FlexCAN protocol engine clock

FlexCAN supports up to an 8 Mbps data rate using a 40 MHz CAN_PE_CLK. With a 16 MHz crystal source, 3.2 Mbps is achievable for baud rate calculations (see the section Protocol timing in the "CAN (FlexCAN)" chapter for details).

For MWCT2016S, the maximum data rate supported in FlexCAN_0 instance is 5.7 Mbps.

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

23.6.1.1.1.1 FlexCAN timestamp implementation

The following figure shows the FlexCAN timestamping implementation. The related table shows the timestamp sources and corresponding clock nodes.

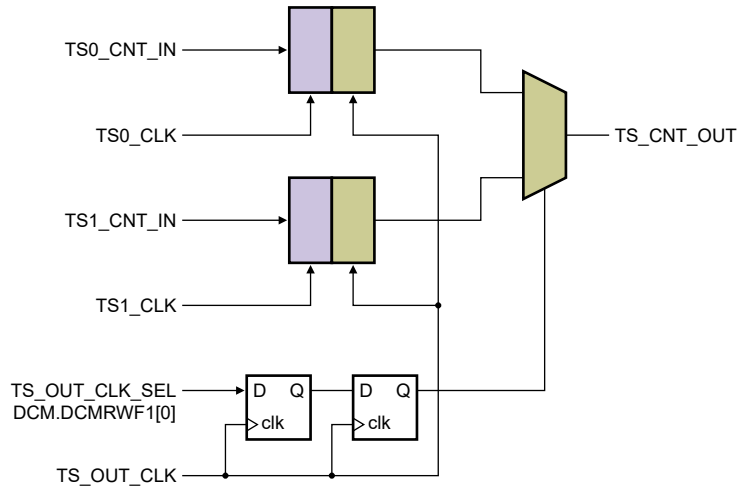


Figure 62. FlexCAN timestamping

Table 102. Timestamp sources and clock nodes

Timestamp	Module	TS clock domain
TS0_CLK	EMAC	TS0_CLK—EMAC_TS_CLK
TS1_CLK	STM0	TS1_CLK—AIPS_PLAT_CLK
TS_OUT_CLK	FlexCAN_*n*	TS_OUT_CLK—EMAC_TS_CLK

NOTE

The timestamp clock (TS_OUT_CLK, EMAC_TS_CLK for MWCT2xxxS) must be greater than or equal to FLEXCAN_*n*_PE_CLK. When using STM0 as the timestamp source, the FlexCAN timestamp clock (TS_OUT_CLK, EMAC_TS_CLK for MWCT2xxxS) must be greater than or equal to STM0_CLK.

23.6.1.1.2 LPI2C_*n* clocking

The following figure shows LPI2C_*n* clocking, and the related table shows the LPI2C SIUL2 configuration.

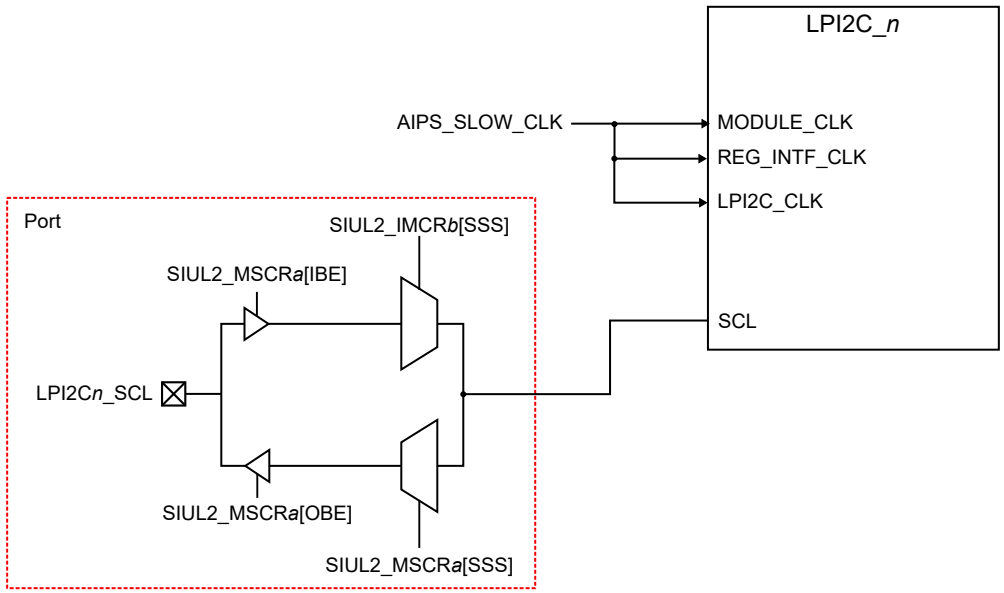


Figure 63. LPI2C clocking

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

Table 103. SIUL2 options for LPI2C*n*_SCL clock

I²C_*n* signal	I²C mode	I/O signal	Port	MSCR*a*	MSCR fields			IMCR*b*	IMCR[SSS]
					OBE	IBE	SSS		
I2C_0_SCL	Slave	LPI2C0_SCL	PTC8	72	0	1	X	212	0001b
	Master				1	0	001b		X
	Slave		PTD14	110	0	1	X		0010b
	Master				1	0	100b		X
	Slave		PTF20[1]	180	0	1	X		0011b
	Master				1	0	100b		X
I2C_0_SCLS	Slave	LPI2C0_SCLS	PTB1	33	0	1	X	213	0001b
	Master				1	0	001b		X
I2C_1_SCL	Slave	LPI2C1_SCL	PTC7	71	0	1	X	217	0001b
	Master				1	0	011b		X
	Slave		PTC15	79	0	1	X		0110b
	Master				1	0	111b		X
	Slave		PTC28	92	0	1	X		0100b

Table continues on the next page...

Table 103. SIUL2 options for LPI2C*n*_SCL clock (continued)

I²C_*n* signal	I²C mode	I/O signal	Port	MSCR*a*	MSCR fields			IMCR*b*	IMCR[SSS]
					OBE	IBE	SSS		
	Master				1	0	101b		X
	Slave		PTD9	105	0	1	X		0010b
	Master				1	0	010b		X
	Slave		PTF7[2]	167	0	1	X		0101b
	Master				1	0	011b		X
I2C_1_SCLS	Slave	LPI2C1_SCLS	PTC17	81	0	1	X	218	0010b
	Master				1	0	100b		X

1. PTF20 pad is not present in MWCT2015S and MWCT2016S.
2. PTF7 pad is not present in MWCT2015S and MWCT2016S.

23.6.1.1.3 EMAC clocking

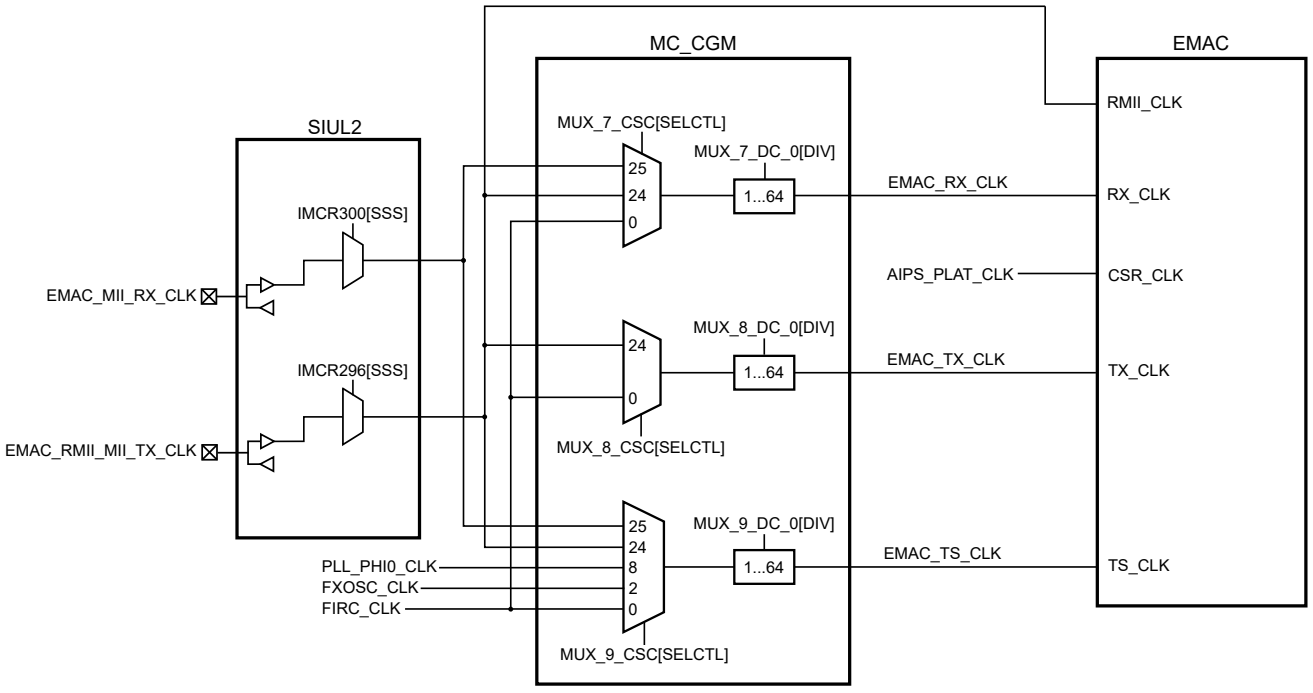


Figure 64. EMAC clocking

NOTE

EMAC operates only in Clock options A and B, since the module clock becomes lower than the protocol clock (RMII/MII clocks) in other modes.

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

PLL_AUX should operate in integer mode to meet the RGMII clock accuracy requirements.

PLL_AUX will be a clock source for the GMAC TX and TS clocks, to support 1 GBps Ethernet operation.

23.6.1.1.3.1 EMAC RMII clocking

The following table shows the EMAC RMII clocking, and the related table shows the SIUL2 clock signal configuration for RMII.

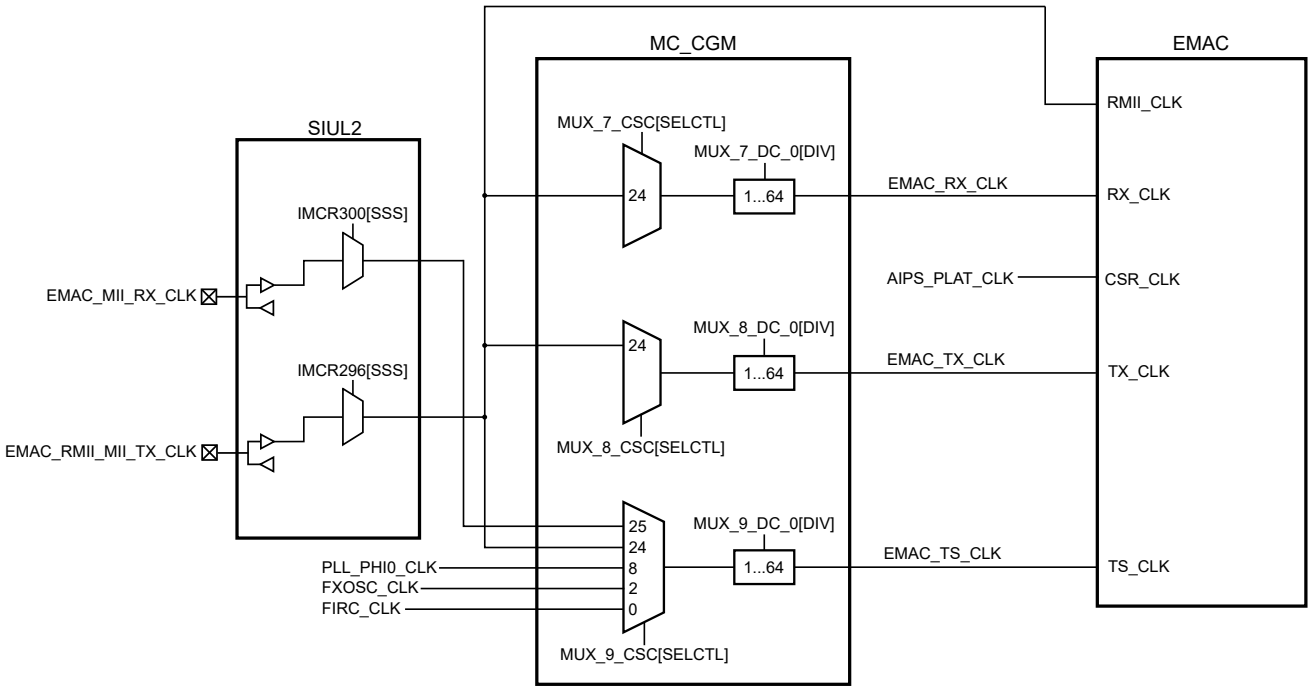


Figure 65. EMAC RMII clocking

Table 104. EMAC RMII clock configuration

Source clock	Destination clock	Port	SIUL2					
			MSCR*a*	MSCR fields			IMCR*b*	IMCR[SSS]
				OBE	IBE	SSS		
EMAC_RMII_MII_TX_CLK	EMAC_RX_CLK	PTC0	64	0	1	X	296	0100b
	EMAC_TX_CLK							
	EMAC_TX_CLK	PTD6	102	0	1	X		0010b
	EMAC_RX_CLK							
	EMAC_TX_CLK	PTD11	107	0	1	X		0001b
	EMAC_RX_CLK							
	EMAC_TX_CLK	PTD12	108	0	1	X		0011b

Table continues on the next page...

Table 104. EMAC RMII clock configuration (continued)

Source clock	Destination clock	Port	SIUL2					
			MSCR*a*	MSCR fields			IMCR*b*	IMCR[SSS]
				OBE	IBE	SSS		
	EMAC_RX_CLK							
EMAC_RMII_MII_TX_CLK (MC_CGM.MUX_9_CSC[SELCTL] = 18h)	EMAC_TS_CLK	PTC0	64	0	1	X	296	0100b
		PTD6	102	0	1	X		0010b
		PTD11	107	0	1	X		0001b
		PTD12	108	0	1	X		0011b
EMAC_MII_RX_CLK (MC_CGM.MUX_9_CSC[SELCTL] = 19h)	EMAC_TS_CLK	PTC1	65	0	1	X	300	0011b
		PTD5	101	0	1	X		0010b
		PTD10	106	0	1	X		0001b

NOTE

The EMAC time-stamp clock (EMAC_TS_CLK) can use either of the source clocks show in the table above, but can also select other clocks as selected in MC_CGM.MUX_9_CSC[SELCTL].

23.6.1.1.3.2 EMAC MII clocking

The following figure shows the EMAC MII clocking, and the related table shows the SIUL2 clock signal configuration for MII.

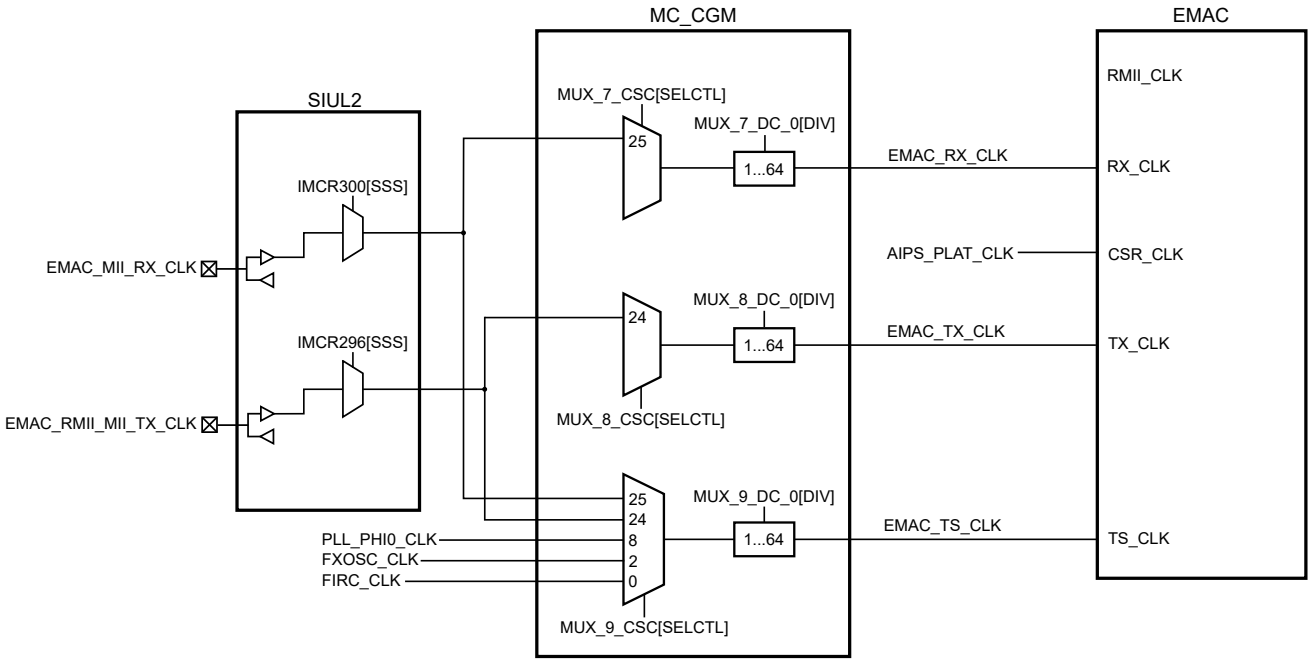


Figure 66. EMAC MII clocking

Table 105. SIUL2 EMAC MII clock configuration

Source clock	Destination	Port	SIUL2					
			MSCR*a*	MSCR fields			IMCR*b*	IMCR[SSS]
				OBE	IBE	SSS		
EMAC_RMII_MII_TX_CLK	EMAC_TX_CLK	PTC0	64	0	1	X	296	0100b
		PTD6	102	0	1	X		0010b
		PTD11	107	0	1	X		0001b
		PTD12	108	0	1	X		0011b
EMAC_MII_RX_CLK	EMAC_RX_CLK	PTC1	65	0	1	X	300	0011b
		PTD5	101	0	1	X		0010b
		PTD10	106	0	1	X		0001b
EMAC_RMII_MII_TX_CLK (MC_CGM.MUX_9_CSC[SELCTL] = 18h)	EMAC_TS_CLK	PTC0	64	0	1	X	296	0100b
		PTD6	102	0	1	X		0010b
		PTD11	107	0	1	X		0001b
		PTD12	108	0	1	X		0011b
EMAC_MII_RX_CLK (MC_CGM.MUX_9_CSC[SELCTL] = 19h)	EMAC_TS_CLK	PTC1	65	0	1	X	300	0011b
		PTD5	101	0	1	X		0010b
		PTD10	106	0	1	X		0001b

NOTE

The EMAC time-stamp clock (EMAC_TS_CLK) can use any of the signals shown in the figure above which includes either EMAC_RX_CLK or EMAC_RMII_MII_TX_CLK source clocks. MC_CGM.MUX_9_CSC[SELCTL] selects the specific source clock for EMAC_TS_CLK.

23.6.1.1.4 LPSPI*n* clocking

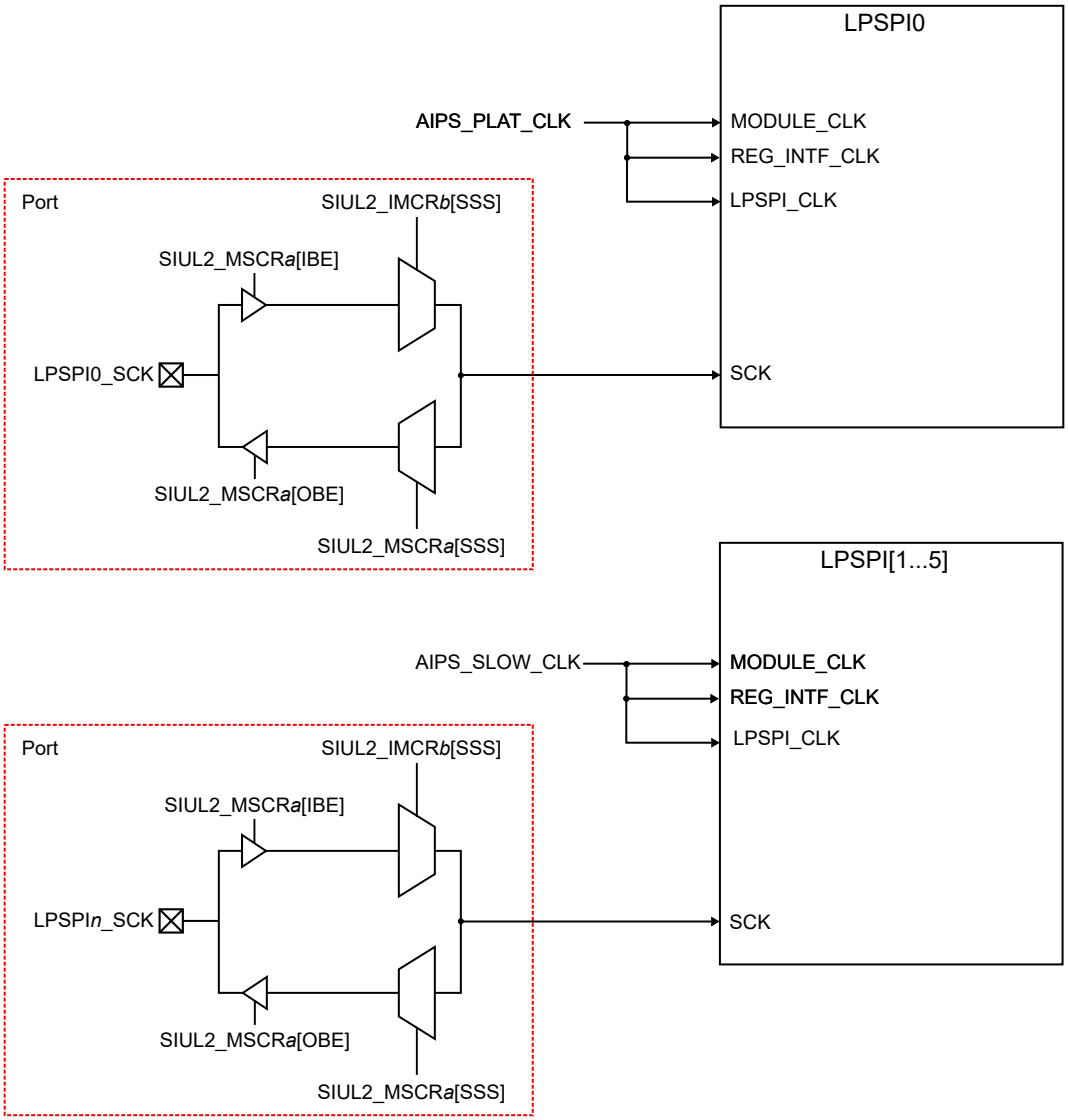


Figure 67. LPSPI*n* clocking

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

Table 106. SIUL2 options for LPSPI*n* SCK

Source	Destination	SPI mode	Port	MSCR *a*	MSCR fields			IMCR *b*	IMCR[SSS]
					OBE	IBE	SSS		
LPSPI0_SCK	LPSPI0 SCK	Slave	PTC8	72	0	1	X	229	0001b
		Master			1	0	0110b		X

Table continues on the next page...

Table 106. SIUL2 options for LPSPI*n* SCK (continued)

Source	Destination	SPI mode	Port	MSCR *a*	MSCR fields			IMCR *b*	IMCR[SSS]
					OBE	IBE	SSS		
		Slave	PTD11	107	0	1	X		0101b
		Master			1	0	0110b		X
		Slave	PTD15	111	0	1	X		0011b
		Master			1	0	0100b		X
		Slave	PTE1	129	0	1	X		0010b
		Master			1	0	0010b		X
LPSPI1_SCK	LPSPI1 SCK	Slave	PTA3	3	0	1	X	238	0001b
		Master			1	0	0011b		X
		Slave	PTA19	19	0	1	X		0011b
		Master			1	0	0100b		X
		Slave	PTA28	28	0	1	X		0100b
		Master			1	0	0011b		X
		Slave	PTB14	46	0	1	X		0010b
		Master			1	0	0011b		X
LPSPI2_SCK	LPSPI2 SCK	Slave	PTB29	61	0	1	X	245	0011b
		Master			1	0	0101b		X
		Slave	PTC15	79	0	1	X		0010b
		Master			1	0	0011b		X
		Slave	PTE15	143	0	1	X		0001b
		Master			1	0	0011b		X
		Slave	PTF0	160	0	1	X		0100b
		Master			1	0	011b		X
LPSPI3_SCK	LPSPI3 SCK	Slave	PTC17	81	0	1	X	252	0011b
		Master			1	0	0001b		X
		Slave	PTD1	97	0	1	X		0001b
		Master			1	0	0011b		X
		Slave	PTE7	135	0	1	X		0010b
		Master			1	0	0110b		X
		Slave	PTF13	173	0	1	X		0100b
		Master			1	0	0100b		X

Table continues on the next page...

Table 106. SIUL2 options for LPSPI*n* SCK (continued)

Source	Destination	SPI mode	Port	MSCR *a*	MSCR fields			IMCR *b*	IMCR[SSS]
					OBE	IBE	SSS		
LPSPI4_SCK	LPSPI4 SCK	Slave	PTB10	42	0	1	X	259	0010b
		Master			1	0	0001b		X
		Slave	PTC27	91	0	1	X		0001b
		Master			1	0	0111b		X
		Slave	PTE22	150	0	1	X		0011b
		Master			1	0	0110b		X
LPSPI5_SCK	LPSPI5 SCK	Slave	PTA3	3	0	1	X	266	0010b
		Master			1	0	0111b		X
		Slave	PTD14	110	0	1	X		0001b
		Master			1	0	0001b		X
		Slave	PTD26	122	0	1	X		0011b
		Master			1	0	0110b		X

23.6.1.1.5 LPUART clocking

The following figure shows the LPUART clocking configuration, and the related table shows LPUART use case baud rates.

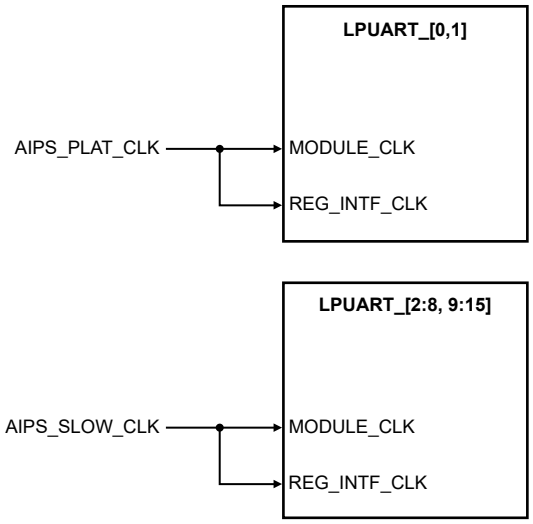


Figure 68. LPUART clocking

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

LPUART_0 and LPUART_1 is clocked by AIPS_PLAT_CLK, to support up to 12 Mbps while the core is running at 120 MHz.

Table 107. LPUART baud rate calculation

Required baud rate (bps)	LPUART_CLK (MHz)	OSR	SBR[12:0]	Calculated baud rate (bps) [1]
8192	40	4	976	8196
8192	80	15	610	8196
8192	48	15	366	8196
115200	48	15	26	115384
115200	40	7	43	116279
19200	80	4	833	19207
19200	40	7	260	19230
38400	40	7	130	38461

1. MODULE_CLK ÷ (LPUART.BAUD[SBR] × (LPUART.BAUD[OSR] + 1))

23.6.1.1.6 FlexIO clocking

The following figure shows the FlexIO clocking interface. The related two tables show the FlexIO baud rate use cases.

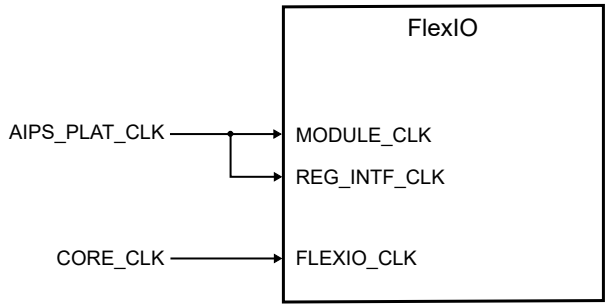


Figure 69. FlexIO clocking

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

Table 108. FlexIO baud rate calculation (FlexIO.TIMCFG*n*[TIMDEC] = 101b)

FLEXIO_CLK (CORE_CLK)	Required baud rate	TIMCMP*n*[CMP]		Theoretical baud rate [1]	Bit duration	Observed baud rate
		Hex	Decimal			
88 MHz	9600	0010h	16	10110.29	101.33 µs	9868
88 MHz	19200	0007h	7	21484.37	47.44 µs	21079
88 MHz	57600	0001h	1	85937.50	11.66 µs	85763
88 MHz	115200	0000h	0	171875.00	5.88 µs	170068

1. Theoretical baud rate = Frequency ÷ (256 × 2 × (TIMCMP*n*[CMP] + 1))

Table 109. FlexIO baud rate calculation (FlexIO.TIMCFG*n*[TIMDEC] = 100b)

FLEXIO_CLK (CORE_CLK)	Required baud rate	TIMCMP*n*[CMP]		Theoretical baud rate [1]	Bit duration	Observed baud rate
		Hex	Decimal			
88 MHz	9600	—	—	—	—	—
88 MHz	19200	8Eh	142	19230.77	51.88 µs	19275
88 MHz	57600	2Eh	46	58510.64	16.89 µs	59206
88 MHz	115200	16h	22	119565.21	8.44 µs	118483

23.6.1.1.7 QuadSPI clocking

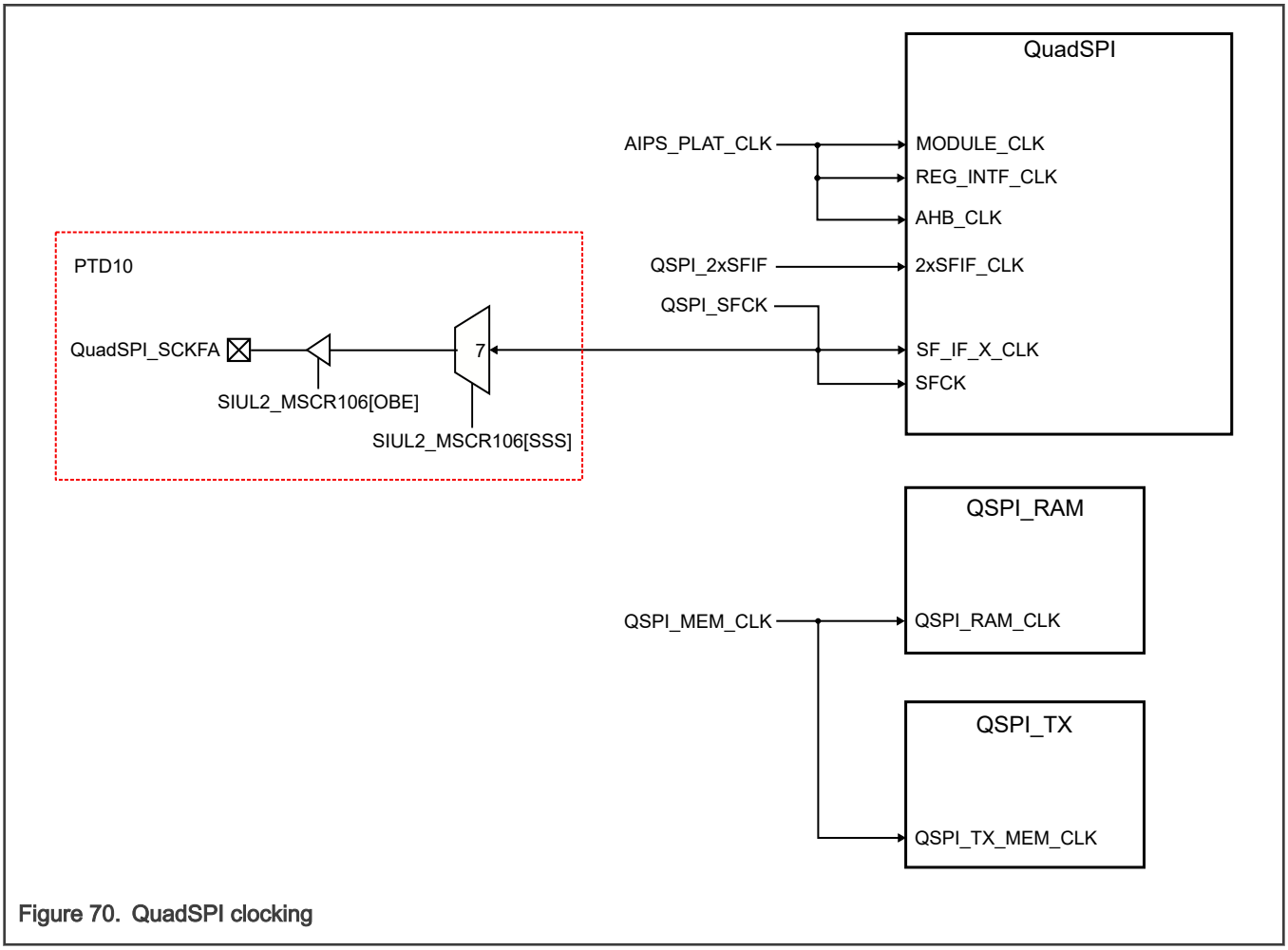


Figure 70. QuadSPI clocking

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

Table 110. QuadSPI clocking

Clock option	AIPS_PLAT_CLK	QSPI_SFCK	QSPI_MEM_CLK
See Option A - High Performance mode (CORE_CLK @ 160 MHz)	80 MHz	120 MHz	160 MHz
See Option B - Reduced Speed mode (CORE_CLK @ 120 MHz)	60 MHz	120 MHz or 80 MHz	120 MHz
See Option F - Operation in 1:1 mode with CORE_CLK and AXBS_CLK at same speed	80 MHz	80 MHz	160 MHz

23.6.1.1.8 SAI_*n* clocking

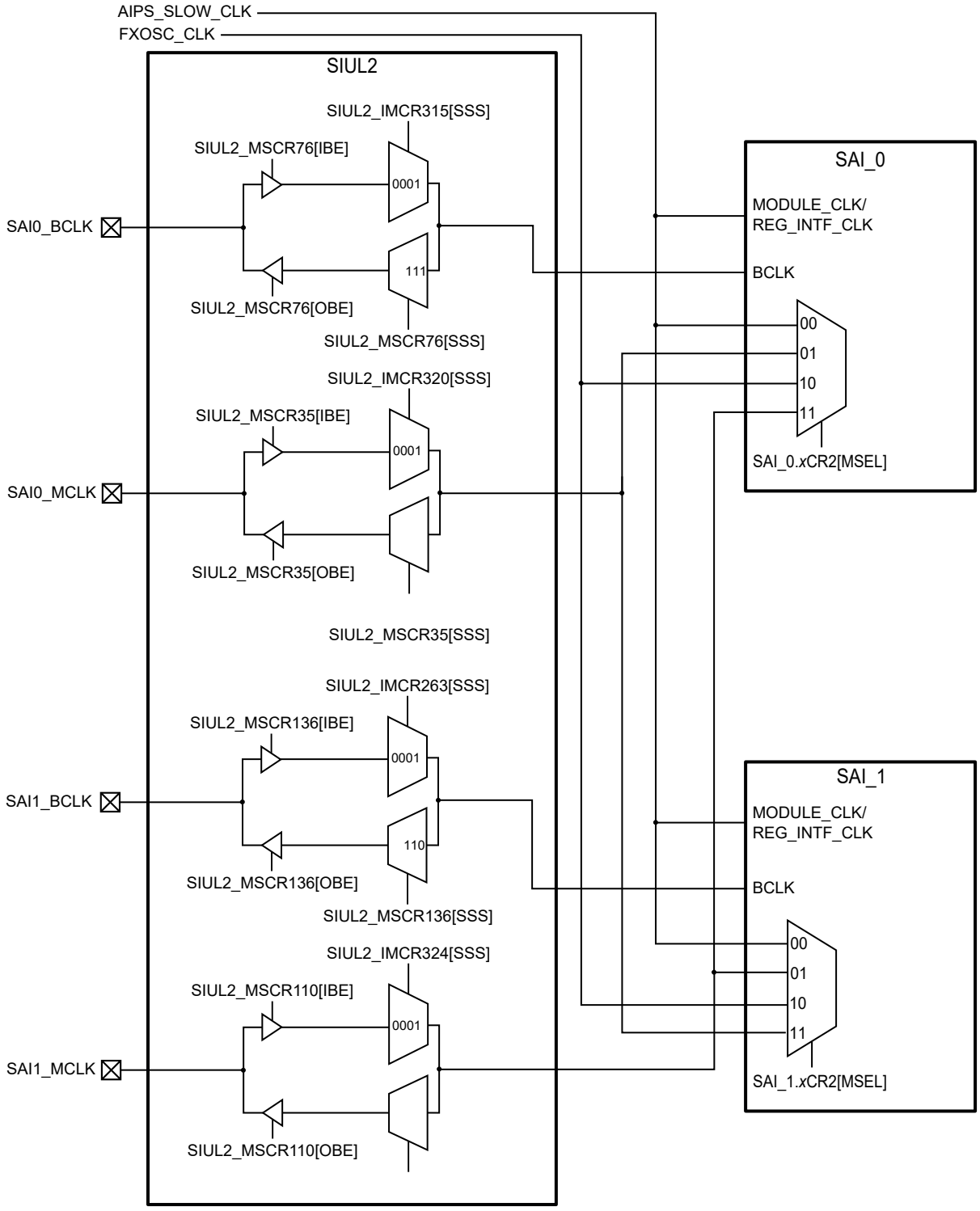


Figure 71. SAI_*n* clocking

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

Table 111. SIUL2 options for SAI_*n*

Source	Destination	Port	MSCR *a*	MSCR fields			IMCR *b*	IMCR[SSS]
				OBE	IBE	SSS		
SAI0_MCLK	SAI_0.*x*CR2[MSEL] = 01b SAI_1.*x*CR2[MSEL] = 11b	PTB3	35	0	1	X	320	0001b
				—	—	Not available		X
SAI1_MCLK	SAI_0.*x*CR2[MSEL] = 11b SAI_1.*x*CR2[MSEL] = 01b	PTD14	110	0	1	X	324	0001b
				1	0	Not available		X
SAI0_BCLK	SAI0 BCLK	PTC12	76	0	1	X	315	0001b
SAI0 BCLK	SAI0_BCLK			1	0	111b		X
SAI1_BCLK	SAI1 BCLK	PTE8	136	0	1	X	322	0001b
SAI1 BCLK	SAI1_BCLK			1	0	110b		X

Internally generated MCLK is not supported on the MWCT2xxxS chip family.

23.6.1.2 System modules

Figure 72 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 112 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

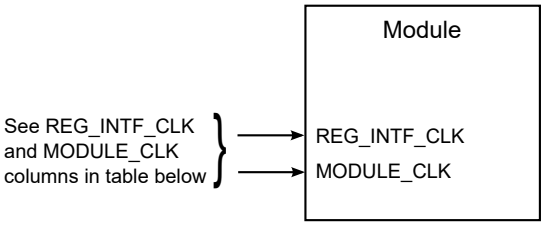


Figure 72. System module clocks

Table 112. System module clocking

Module	MODULE_CLK	REG_INTF_CLK
MSCM	AIPS_PLAT_CLK	AIPS_PLAT_CLK
MCM	AIPS_SLOW_CLK	AIPS_SLOW_CLK

Table continues on the next page...

Table 112. System module clocking (continued)

Module	MODULE_CLK	REG_INTF_CLK
SIUL2	See SIUL2 clocking.	
VIRT_WRAPPER	AIPS_SLOW_CLK	AIPS_SLOW_CLK
AXBS	CORE_CLK	AIPS_PLAT_CLK
DMAMUX	CORE_CLK	CORE_CLK
eDMA	CORE_CLK	AIPS_PLAT_CLK
INTM	AIPS_PLAT_CLK	AIPS_PLAT_CLK
SEMA42	AIPS_PLAT_CLK	AIPS_PLAT_CLK
XBIC	CORE_CLK	AIPS_PLAT_CLK
XRDC	CORE_CLK	AIPS_PLAT_CLK

23.6.1.2.1 SIUL2 clocking

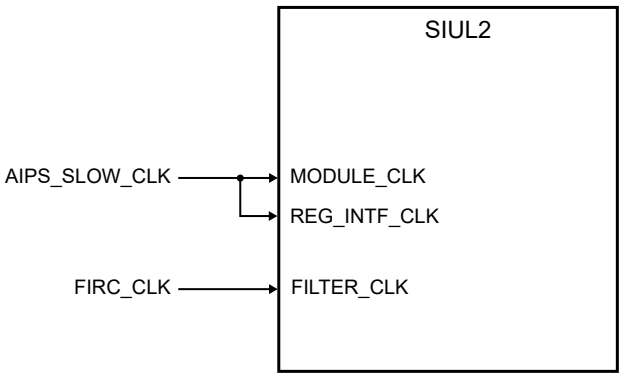


Figure 73. SIUL2 clocking

23.6.1.3 Clocking modules

Figure 74 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 113 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

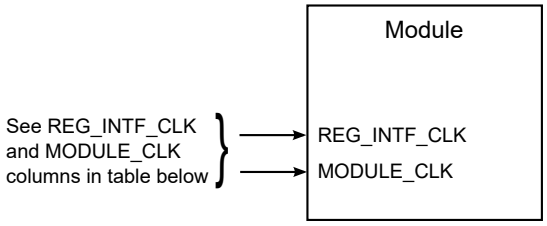


Figure 74. Clocking module clocks

Table 113. Clocking module clocking

Module	MODULE_CLK	REG_INTF_CLK
FXOSC	See FXOSC clocking.	
SXOSC	See SXOSC clocking.	
SIRC	See SIRC clocking.	
FIRC	See FIRC clocking.	
PLLDIG	See PLLDIG clocking.	
MC_CGM	—	AIPS_SLOW_CLK

23.6.1.3.1 FIRC clocking

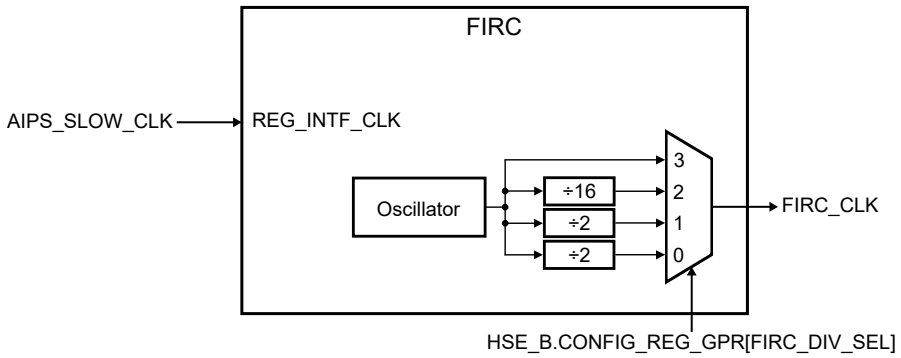


Figure 75. FIRC clocking

23.6.1.3.2 SIRC clocking

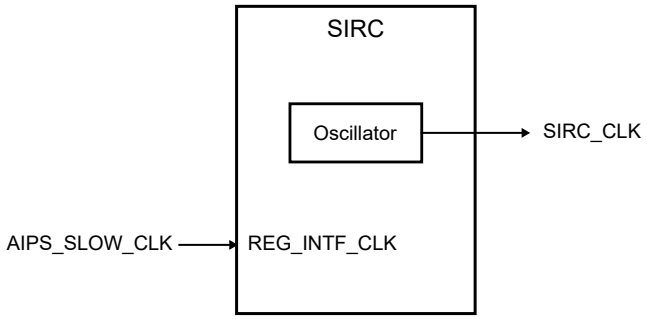


Figure 76. SIRC clocking

23.6.1.3.3 FXOSC clocking

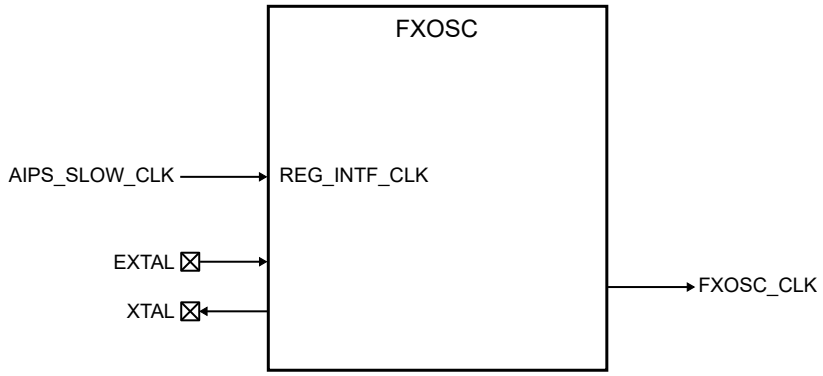


Figure 77. FXOSC clocking

23.6.1.3.4 SXOSC clocking

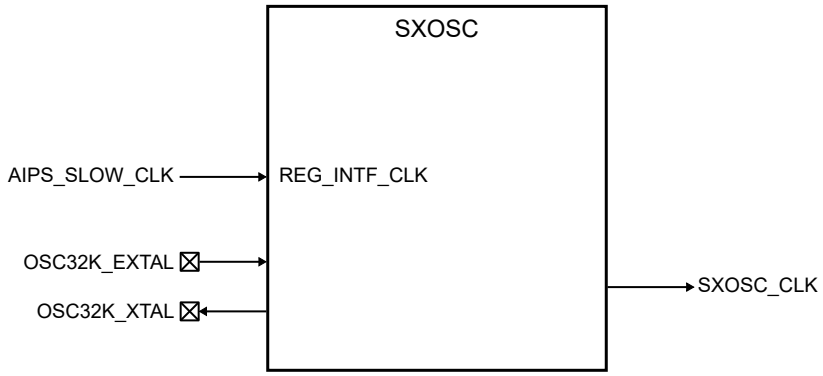


Figure 78. SXOSC clocking

23.6.1.3.5 PLLDIG clocking

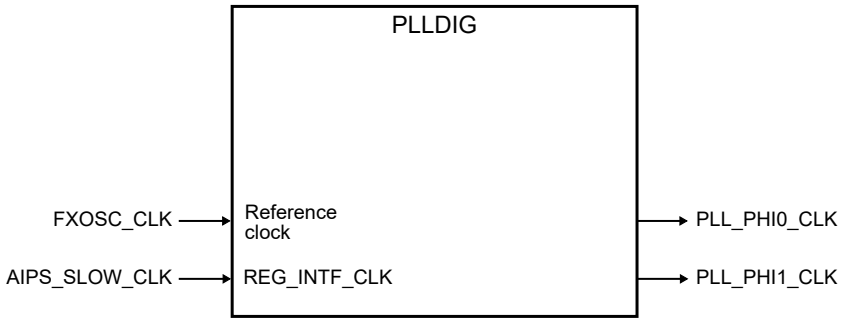


Figure 79. PLLDIG clocking

23.6.1.4 Reset modules

Figure 80 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 114 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

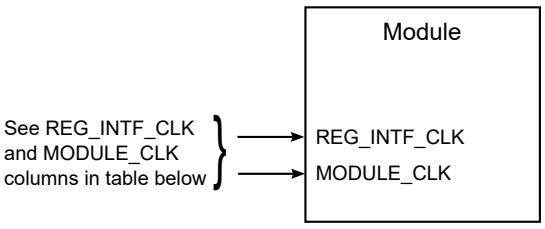


Figure 80. Reset module clocks

Table 114. Reset module clocking

Module	MODULE_CLK	REG_INTF_CLK
MC_RGM	FIRC_CLK	FIRC_CLK

23.6.1.5 Security modules

Figure 81 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 115 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

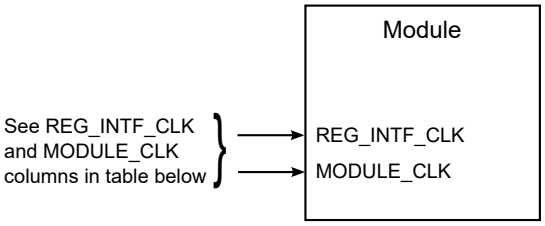


Figure 81. Security module clocks

Table 115. Security module clocking

Module	MODULE_CLK	REG_INTF_CLK
HSE_B	See HSE_B clocking .	
MU_*n*	AIPS_SLOW_CLK	AIPS_SLOW_CLK
DCM	DCM_CLK	DCM_CLK

23.6.1.5.1 HSE_B clocking

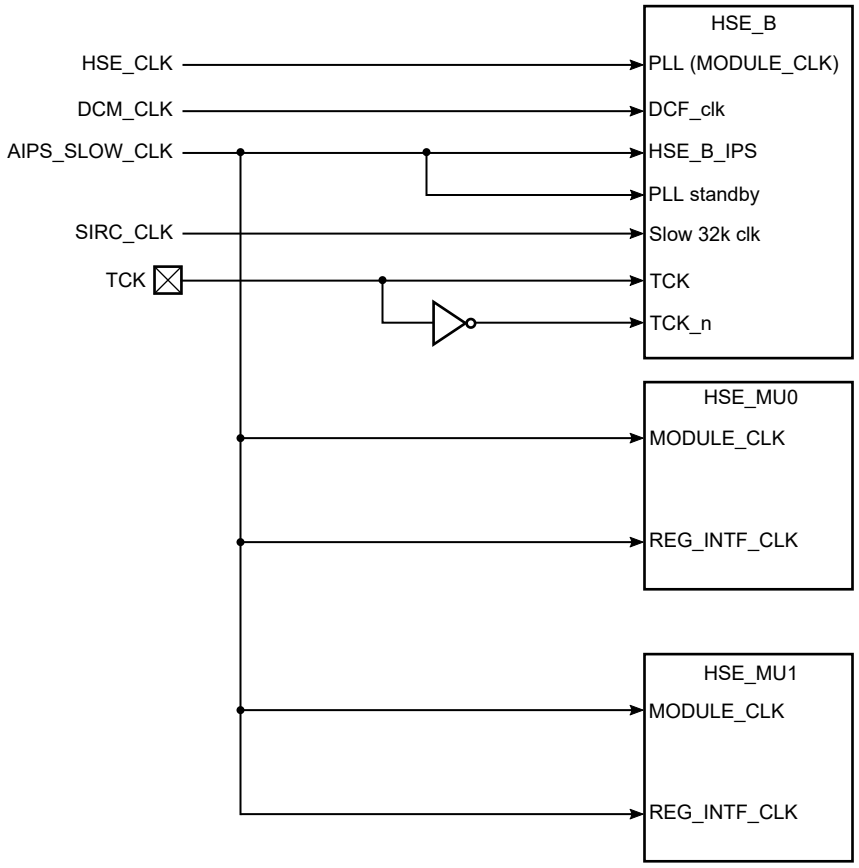


Figure 82. HSE_B clocking

23.6.1.6 Power-management modules

Figure 83 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 116 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

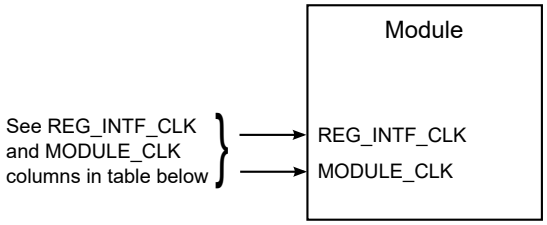


Figure 83. Power-management module clocks

Table 116. Power-management module clocking

Module	MODULE_CLK	REG_INTF_CLK
PMC	AIPS_SLOW_CLK	AIPS_SLOW_CLK
MC_ME	AIPS_SLOW_CLK	AIPS_SLOW_CLK
MC_PCU	FIRC_CLK	FIRC_CLK
WKPU	AIPS_SLOW_CLK	AIPS_SLOW_CLK

23.6.1.7 Safety modules

Figure 84 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 117 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

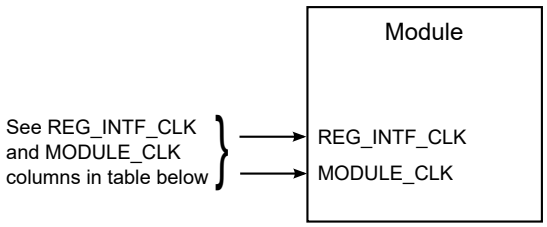


Figure 84. Safety module clocks

Table 117. Safety module clocking

Module	MODULE_CLK	REG_INTF_CLK
EIM	AIPS_PLAT_CLK	AIPS_PLAT_CLK
ERM	See ERM clocking.	
FCCU	See FCCU clocking.	
STCU2	See STCU2 clocking.	
REG_PROT	AIPS_SLOW_CLK	AIPS_SLOW_CLK
CMU_FC	AIPS_SLOW_CLK	AIPS_SLOW_CLK
CMU_FM	AIPS_SLOW_CLK	AIPS_SLOW_CLK
CRC	AIPS_PLAT_CLK	AIPS_PLAT_CLK

23.6.1.7.1 FCCU clocking

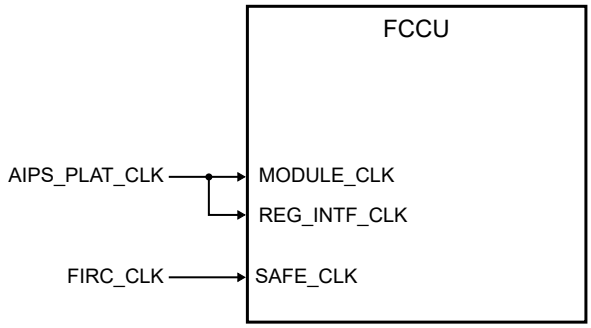


Figure 85. FCCU clocking

23.6.1.7.2 STCU2 clocking

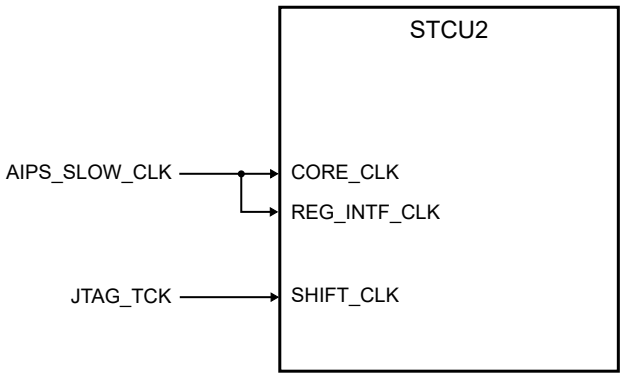


Figure 86. STCU2 clocking

23.6.1.7.3 ERM clocking

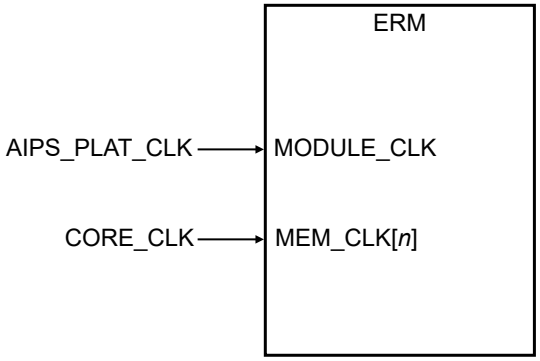


Figure 87. ERM clocking

NOTE

MEM_CLK[20:23] are not used. Source clock for MEM_CLK[0:19] is CORE_CLK.

23.6.1.8 ADC and motor control modules

Figure 88 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 118 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

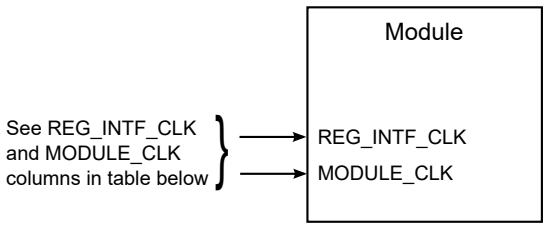


Figure 88. Motor control module clocks

Table 118. Motor control module clocking

Module	MODULE_CLK	REG_INTF_CLK
ADC	See ADC_*n* clocking.	
LCU	CORE_CLK	
eMIOS	See eMIOS_*n* clocking.	
BCTU	See BCTU clocking.	
TRGMUX	AIPS_SLOW_CLK	
TSPC	AIPS_SLOW_CLK	

23.6.1.8.1 ADC_*n* clocking

The following figure shows ADC_*n* clocking configuration.

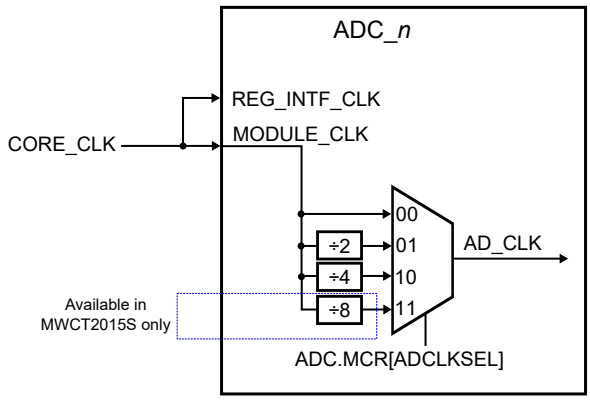


Figure 89. ADC_*n* clocking

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

The prescaler can be bypassed when using FIRC_CLK as the source (see MC_CGM mux 0 clocks for FIRC_CLK use details). The prescaler must be controlled such that the AD_CLK frequency is less than or equal to 120 MHz for MWCT2016S/MWCT2015S. For other MWCT2xxxS products the frequency is 80 MHz.

The minimum operating speed of AD_CLK is 6 MHz using the following configuration:

1. Use FIRC_CLK (48 MHz) as clock source (MC_CGM.MUX_0_CSC[SELCTL] equal to 0000b).
2. Divide FIRC_CLK by 2 for CORE_CLK speed (MC_CGM.MUX_0_DC_0[DIV] equal to 1 (FIRC_CLK divide by 2 = 24 MHz)).
3. Write ADC.MCR[ADCCLKSEL] equal to 10b to divide the FIRC_CLK further by 4 (AD_CLK = 6 MHz).

However, at this lower speed, the ADC_*n* results will be degraded.

23.6.1.8.2 eMIOS_*n* clocking

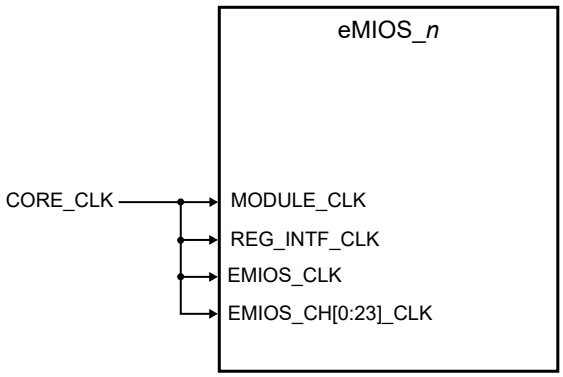


Figure 90. eMIOS_*n* clocking

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

23.6.1.8.3 BCTU clocking

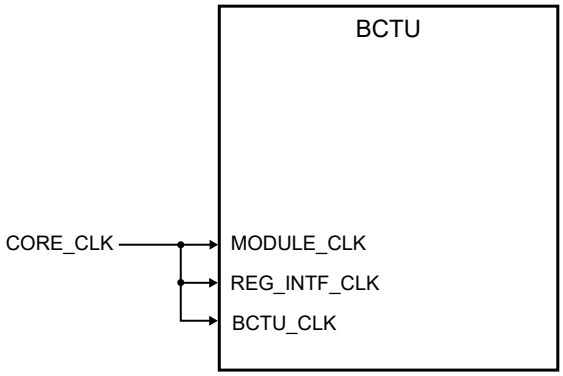


Figure 91. BCTU clocking

23.6.1.9 Timer modules

Figure 92 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 119 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

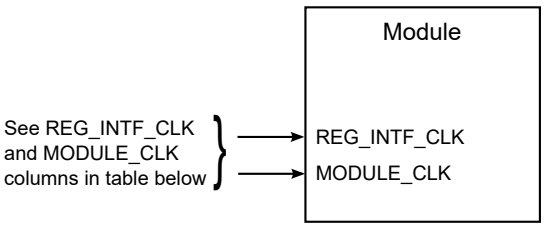


Figure 92. Timer module clocks

Table 119. Timer module clocking

Module	MODULE_CLK	REG_INTF_CLK
PIT	See PIT_*n* clocking.	
SWT	See SWT_*n* clocking.	
STM_*n*	STM0_CLK	STM0_CLK
RTC	See RTC clocking.	

23.6.1.9.1 PIT_*n* clocking

The following figure shows the PIT_*n* clocking configuration. The related tables show the use case configuration.

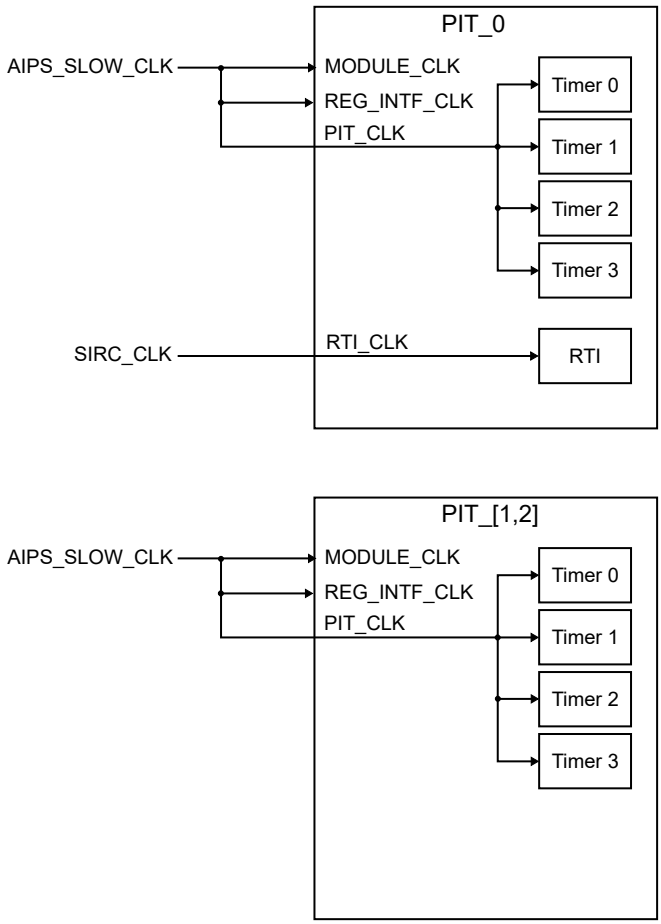


Figure 93. PIT_*n* clocking

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

Table 120. PIT_0 modes of operation

MC_ME.PRTN0_COFB1_CLKEN[REQ44]	PIT_0.MCR[MDIS]	PIT_0.MCR[MDIS_RTI]	Mode	Application
0	X	X	PIT_0 clock gated (minimum power)	Module clock gated and unused / Standby mode, PIT and RTI unused
1	0	0	Both PIT and RTI enabled	Run mode
1	0	1	PIT running, RTI disabled	Run mode with only PIT active
1	1	0	PIT disabled, RTI enabled	Standby mode with RTI enabled
1	1	1	Both PIT and RTI disabled	Standby mode with RTI disabled

Table 121. PIT_[1,2] modes of operation

MC_ME.PRTN*n*_COFB1_CLKEN[REQ[45,63]] [1]	PIT_[1,2].MCR[MDIS]	Mode	Application
0	X	PIT_[1,2] clock gated (minimum power)	Module clock gated and unused, Standby mode
1	0	PIT_[1,2] enabled	Run mode
1	0	PIT_[1,2] running	Run mode with PIT active
1	1	PIT_[1,2] disabled	Standby mode

1. PIT_1 and PIT_2 MC_ME partition registers used are:
   - PIT_1 - MC_ME.PRTN0_COFB1_CLKEN[REQ45]
   - PIT_2 - MC_ME.PRTN1_COFB1_CLKEN[REQ63]

23.6.1.9.2 SWT_*n* clocking

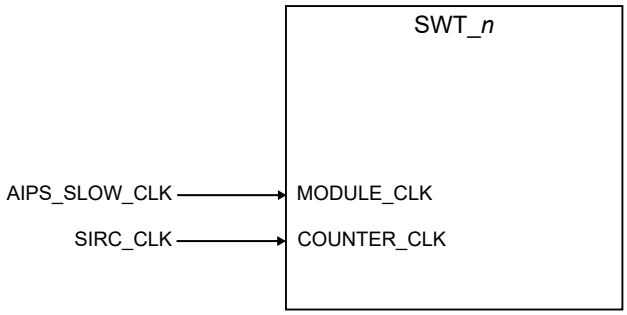


Figure 94. SWT_*n* clocking

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

23.6.1.9.3 RTC clocking

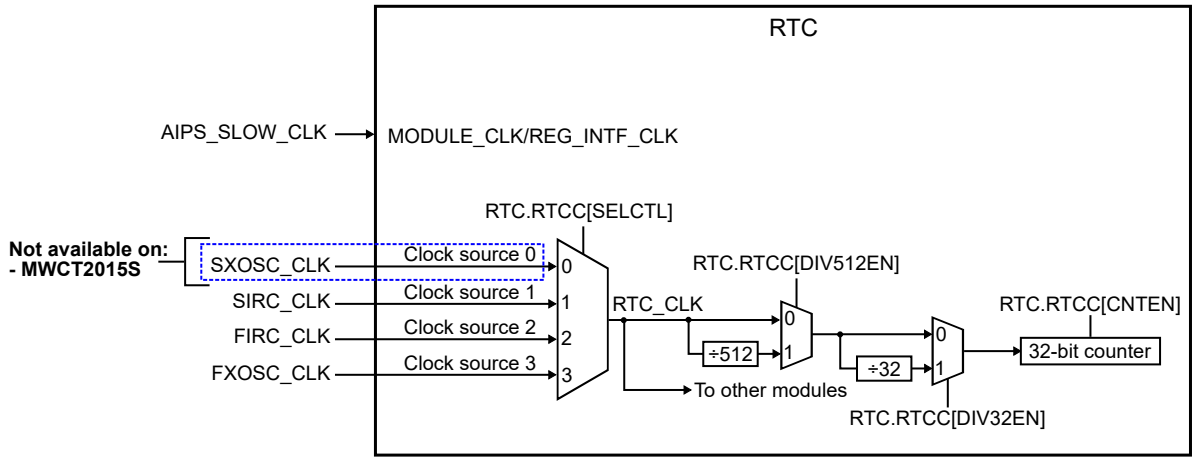


Figure 95. RTC clocking

NOTE

The RTC is available in Standby mode. Although bus clock is gated, the RTC can run on FIRC_CLK, SIRC_CLK, FXOSC_CLK, or SXOSC_CLK.

23.6.1.10 Debug modules

Figure 96 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 122 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

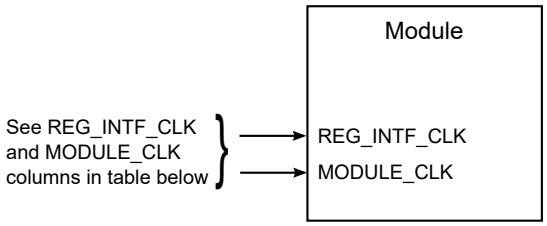


Figure 96. Debug module clocks

Table 122. Debug module clocking

Module	MODULE_CLK	REG_INTF_CLK
JTAGC	See JTAGC clocking.	
JDC	See JDC clocking.	

23.6.1.10.1 JTAGC clocking

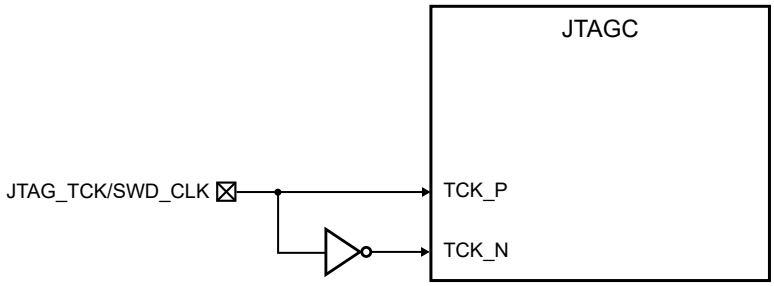


Figure 97. JTAGC clocking

23.6.1.10.2 JDC clocking

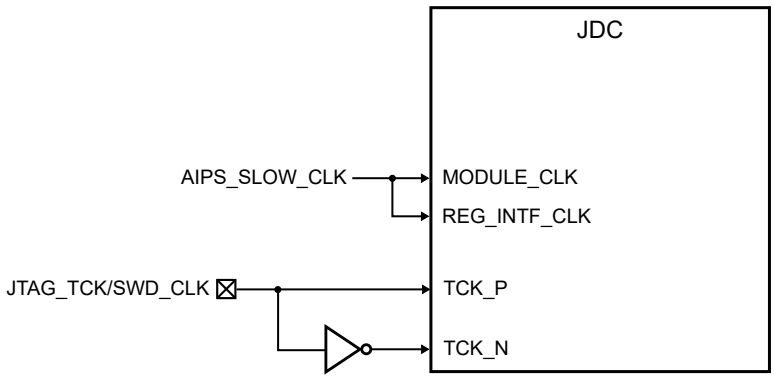


Figure 98. JDC clocking

23.6.1.11 Analog modules

Figure 99 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 123 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

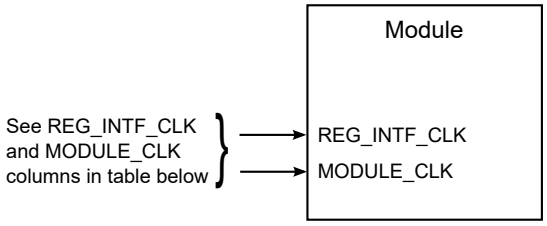


Figure 99. Analog module clocks

Table 123. Analog module clocking

Module	MODULE_CLK	REG_INTF_CLK
LPCMP	See LPCMP_*n* clocking.	
Temperature Sensor	AIPS_SLOW_CLK	AIPS_SLOW_CLK

23.6.1.11.1 LPCMP_*n* clocking

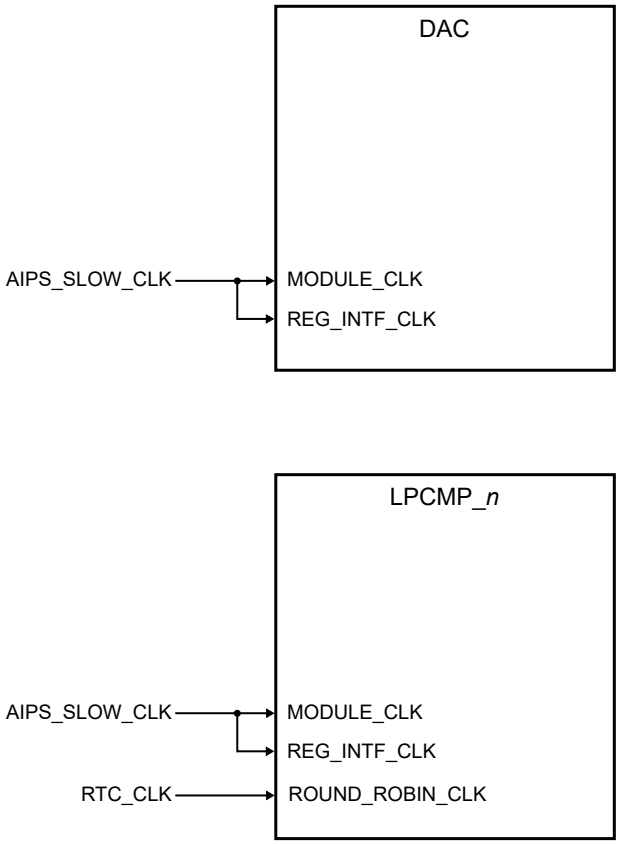


Figure 100. LPCMP_*n* clocking

NOTE

- See Feature comparison for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

23.6.1.12 Memory modules

Figure 101 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 124 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

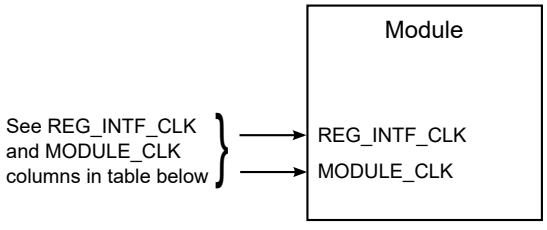


Figure 101. Memory module clocks

Table 124. Memory module clocking

Module	MODULE_CLK	REG_INTF_CLK
PFLASH/ FLASH	CORE_CLK	AIPS_SLOW_CLK
PRAM/ SRAM	CORE_CLK	AIPS_SLOW_CLK

23.6.2 Peripheral data rates

Table 125. Peripheral data rates

Peripheral	Maximum data rate	
	MWCT2D16S and MWCT2D17S	MWCT2015S and MWCT2016S
ADC	See the *MWCT2xxxS Data Sheet* for details.	See the *MWCT2xxxS Data Sheet* for details.
eMIOS[1]	Able to shift PWM edge by 1 ÷ (160 MHz) = 6.25 ns.	Able to shift PWM edge by 1 ÷ (120 MHz) = 8.33 ns.
BCTU[2]	BCTU to generate triggers at 160 MHz.	BCTU to generate triggers at 120 MHz.
LCU	Same domain as EMIOS and BCTU.	Same domain as EMIOS and BCTU.
QuadSPI	• Flash memory interface: SDR 120 MHz • DDR and hyperflash not supported	• Flash memory interface: SDR 120 MHz • DDR and hyperflash not supported
EMAC	See the section EMAC instance and configuration in the "Ethernet Media Access Controller (EMAC)" chapter for details.	See the section EMAC instance and configuration in the "Ethernet Media Access Controller (EMAC)" chapter for details.
FlexCAN[3]	8 Mbps	8 Mbps
LPI2C[4]	400 Kbps in fast mode.	400 Kbps in fast mode.
LPSPI[5]	• LPSPI0 is to have a high clock rate of 20 Mbps • LPSPI1–LPSPI5 can be 10 Mbps	• LPSPI0 is to have a high clock rate of 15 Mbps • LPSPI1–LPSPI3 can be 7.5 Mbps
SAI0/SAI1 (I2S)[6]	• Bit rate = 12.288 MHz (12.288 Mbps—bit clock frequency governs the bit rate) • Master clock = 24.576 MHz • SAI0 and SAI1 operate asynchronously to each other	• Bit rate = 12.288 MHz (12.288 Mbps—bit clock frequency governs the bit rate) • Master clock = 24.576 MHz • SAI0 and SAI1 operate asynchronously to each other
LPUART[7]	See the section Baud rate generation in the "Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)" chapter and Table 107 for details	See the section Baud rate generation in the "Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)" chapter and Table 107 for details
FlexIO[8]	The different protocol data rates supported by FlexIO are listed below. For master mode, max baud rate is FLEXIO_CLK ÷ 4. For slave mode, max baud rate is FLEXIO_CLK ÷ 6. The baud rate is controlled by TIMCMP (lower 8 bits in 8-bit mode and 16 bits in 16-bit mode). • UART: 19200 bps	The different protocol data rates supported by FlexIO are listed below. For master mode, max baud rate is FLEXIO_CLK ÷ 4. For slave mode, max baud rate is FLEXIO_CLK ÷ 6. The baud rate is controlled by TIMCMP (lower 8 bits in 8-bit mode and 16 bits in 16-bit mode). • UART: 19200 bps

Table continues on the next page...

Table 125. Peripheral data rates (continued)

Peripheral	Maximum data rate	
	MWCT2D16S and MWCT2D17S	MWCT2015S and MWCT2016S
	• I2C: 400 Kbps • SPI: 10 Mbps • I2S: 12.288 Mbps	• I2C: 400 Kbps • SPI: 7.5 Mbps • I2S: 12.288 Mbps
Trace	• Fast-speed pins: 120 MHz • Medium-speed pins: 48 MHz	• SWO trace

1. See section Global clock prescaler (GCP) in the "Enhanced Modular IO Subsystem (eMIOS)" chapter for details.
2. See section Triggers in the "Enhanced Modular IO Subsystem (eMIOS)" chapter.
3. See the section Protocol timing in the "CAN (FlexCAN)" chapter for data rate calculation details.
4. See the section Clocks in the "Low Power Inter-Integrated Circuit (LPI2C)" chapter for LPI2C_CLK frequency details.
5. See the section Clocks in the "Low Power Serial Peripheral Interface (LPSPI)" chapter.
6. See the section SAI clocking in the "Synchronous Audio Interface (SAI)" chapter and SAI_*n* clocking for details.
7. At least one pair of LPUART instances (LPUART0 and LPUART1) support up to 12 Mbps.
8. See the section Application Information and FlexIO instances and configuration in the "Flexible I/O (FlexIO)" chapter and FlexIO clocking for baud configuration details.

23.6.3 Core and peripheral clock control

The chip provides provisions for core and peripheral clock gating. The next sections describe the details on clock gating possibilities and controls (see Power Management Controller (PMC) and Mode Entry Module (MC_ME) for details).

23.6.3.1 Clock gating

Application core clocks are gated by individual MC_ME core clock enable bits. Additionally, application cores can be clock gated by executing WFI (see the "Mode Entry Module (MC_ME)" chapter for details).

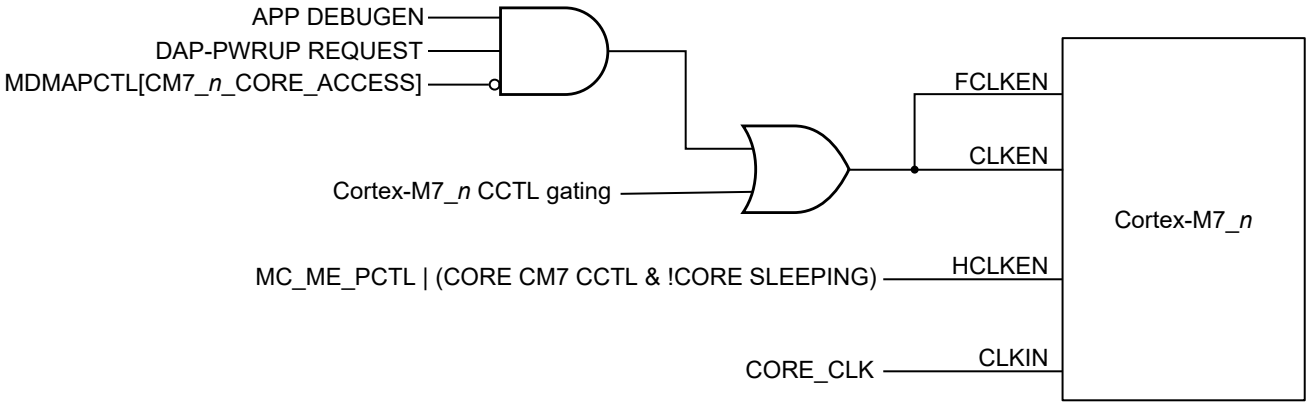


Figure 102. Cortex-M7 core clock gating

To support core debug across functional reset, challenge response done (application debug enable) gating is done so that the debugger can access core debug logic. When the debugger completes its programming with core debug logic, it must program the MDM_AP DAP control bit to shift the control of CLKIN and FCLK to CCTL.

There are two cases in which other masters can access the TCM of each core:

- When the TCM is used as system memory—HCLK will always remain on if TCM PCTL is 1.

- In applications where TCM is not used as system memory, TCM PCTL is written to 0. TCM will then function as the core's memory and HCLK will be gated on WFI.

See the section "MDM_AP register descriptions" in the Debug Subsystem chapter and the Memory Map chapter for details.

23.6.3.2 Peripheral clock gating

See the tables in section "Peripheral clock gating" for the chip partitions, plus peripheral initialization and shutdown details.

23.7 Clocking details

23.7.1 System clock frequency limitations

Table 126. System clock frequency limitations

System clock node	System clock divider	Maximum frequency allowed	Remarks
CORE_CLK	MC_CGM.MUX_0_DC_0[DIV]	• 160 MHz: MWCT2D17S and MWCT2D16S • 120 MHz: MWCT2015S and MWCT2016S	CORE_CLK is always greater than or equal to AIPS_PLAT_CLK.
AIPS_PLAT_CLK	MC_CGM.MUX_0_DC_1[DIV]	• 80 MHz: MWCT2D17S and MWCT2D16S • 60 MHz: MWCT2015S and MWCT2016S	AIPS_PLAT_CLK is always less than or equal to CORE_CLK.
AIPS_SLOW_CLK	MC_CGM.MUX_0_DC_2[DIV]	• 40 MHz: MWCT2D17S and MWCT2D16S • 30 MHz: MWCT2015S and MWCT2016S	AIPS_SLOW_CLK is always less than or equal to AIPS_PLAT_CLK.
HSE_CLK	MC_CGM.MUX_0_DC_3[DIV]	• 120 MHz	When CORE_CLK is equal to or less than 120 MHz, HSE_CLK can be equal to CORE_CLK. When CORE_CLK is higher than 120 MHz, HSE_CLK must be half of the CORE_CLK.
DCM_CLK	MC_CGM.MUX_0_DC_4[DIV]	• 48 MHz	DCM_CLK
LBIST_CLK	MC_CGM.MUX_0_DC_5[DIV]	• 48 MHz: MWCT2D17S, MWCT2D16S,MWCT2015S, and MWCT2016S	LBIST clock
QSPI_MEM_CLK	MC_CGM.MUX_0_DC_6[DIV]	• 160 MHz	QSPI_MEM_CLK is always equal to CORE_CLK except in 1:1 mode (see Option F - Operation in 1:1 mode with CORE_CLK and AXBS_CLK at same speed).

NOTE

The chip supports 1:1 clocking mode, whereby the core(s) are clocked at the same frequency as the slave ports (flash memory, PRAM controller, AIPS controller). Option F - Operation in 1:1 mode with CORE_CLK and AXBS_CLK at same speed supports this requirement.

The frequencies in the table above are maximum frequencies for a specific clock. However, any clock frequency selected must adhere to the same clock divider ratios shown in Clocking use case examples.

23.7.2 Clocking use case examples

The chip supports the clocking modes shown in the subsequent sections, as follows:

- Option A - High Performance mode (CORE_CLK @ 160 MHz) (only available in Run mode)
- Option B - Reduced Speed mode (CORE_CLK @ 120 MHz) (only available in Run mode)
- Option C - Boot Standby mode (CORE_CLK @ 24 MHz)
- Option D - Low-Speed RUN mode (CORE_CLK @ 48 MHz)
- Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz)
- Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 kHz)
- Option F - Operation in 1:1 mode with CORE_CLK and AXBS_CLK at same speed (only available in Run mode)

The following list are requirements when configuring the clocking system.

- Configure PRAM/SRAM wait states and flash memory read/write wait cycles in clock configurations where CORE_CLK equals AIPS_PLAT_CLK. See "Option B" in Gasket configurations in various clocking modes for PRAM/SRAM wait states and gasket configurations for clocking options.
- Enable PLL_PHI*n*_CLK only if using it as the system clock source.
- Disable PLL_PHI*n*_CLK when using FIRC_CLK as the system clock, you must.
- Configure the PMC last mile regulator before enabling the PLL. Configure PMC.CONFIG[LMEN] and PMC.CONFIG[LMBCTLEN] (if using an external BJT) to enable the last mile regulator.
- Disable the PLL before disabling the last mile regulator (see PMC last-mile regulator auto-enable feature for last mile regulator details).

23.7.2.1 Option A - High Performance mode (CORE_CLK @ 160 MHz)

Table 127. Option A - High Performance mode (CORE_CLK @ 160 MHz)

Clocking options	Clock frequencies	
	MWCT2D17S and MWCT2D16S [1]	
PLL VCO frequency	960 MHz	
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	480 MHz (02h)	
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	—	
PLL_PHI1_CLK-related clocks [2]		
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	240 MHz (0001b)	160 MHz (0010b)

Table continues on the next page...

Table 127. Option A - High Performance mode (CORE_CLK @ 160 MHz) (continued)

Clocking options	Clock frequencies	
	MWCT2D17S and MWCT2D16S [1]	
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	120 MHz (0001b)	80 MHz (0001b)
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	For fast pads	
	120 MHz (0001b)	80 MHz (0001b)
	For medium pads	
	48 MHz (0100b)	40 MHz (011b)
PLL_PHI0_CLK-related clocks [3]		
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	160 MHz (0010b)	
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	160 MHz (0000b)	
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	160 MHz (0000b)	
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	80 MHz (0001b)	
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	40 MHz (0011b)	
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	40 MHz (0011b)	
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	80 MHz (0001b)	
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	40 MHz (0011b)	

1. This table does not apply to MWCT2015S or MWCT2016S.
2. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1001b.
3. MC_CGM.MUX_0_CSC[SELCTL] must equal 1000b.

23.7.2.2 Option B - Reduced Speed mode (CORE_CLK @ 120 MHz)

Table 128. Option B - Reduced Speed mode (CORE_CLK @ 120 MHz)

Clocking options	Clock frequencies		
	MWCT2D17S and MWCT2D16S		MWCT2015S and MWCT2016S
PLL VCO frequency	960 MHz		960 MHz
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	480 MHz (02h)		240 MHz (04h)
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	—		
PLL_PHI1_CLK-related clocks [1], [2]			
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	240 MHz (0001b)	160 MHz (0010b)	48 MHz) (0100b)
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	120 MHz (001b)	80 MHz (001b)	—
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	**For fast pads**		—
	120 MHz (001b)	80 MHz (001b)	—
	For medium pads		—
	48 MHz (100b)	40 MHz (011b)	—
PLL_PHI0_CLK-related clocks [3]			
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	120 MHz (011b)		120 MHz (001b)
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	120 MHz (000b)		120 MHz (000b)
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	120 MHz (000b)		—

Table continues on the next page...

Table 128. Option B - Reduced Speed mode (CORE_CLK @ 120 MHz) (continued)

Clocking options	Clock frequencies			
	MWCT2D17S and MWCT2D16S		MWCT2015S and MWCT2016S	
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	60 MHz (001b)		60 MHz (001b)	
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	30 MHz (011b)		30 MHz (011b)	
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	30 MHz (011b)		30 MHz (011b)	
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	120 MHz (000b)	60 MHz (001b)	120 MHz (000b)	60 MHz (001b)
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	30 MHz (011b)		30 MHz (011b)	

1. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1001b.
2. Do not use the combination of different frequencies mentioned below across the 2x2 matrix. The values shown in each cell are valid and must not be clubbed with the values mentioned in any other cells.
3. MC_CGM.MUX_0_CSC[SELCTL] must equal 1000b.

23.7.2.3 Option C - Boot Standby mode (CORE_CLK @ 24 MHz)

Table 129. Option C - Boot Standby mode (CORE_CLK @ 24 MHz)

Clocking options	Clock frequencies
	MWCT2xxxS
PLL VCO frequency	—
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	—
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	24 MHz[1] (00b)
PLL_PHI1_CLK-related clocks [2]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	—
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	—
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	—

Table continues on the next page...

Table 129. Option C - Boot Standby mode (CORE_CLK @ 24 MHz) (continued)

Clocking options	Clock frequencies
	MWCT2xxxS
PLL_PHI0_CLK-related clocks [3]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	—
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	—
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	—
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	24 MHz (0000b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	24 MHz (0000b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	12 MHz (0001b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	24 MHz (0000b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	24 MHz (0000b)

1. The FIRC_DIV_SEL is configured by the sBAF code. It is set to 11b after reset or normal standby exit and FIRC_CLK is 48 MHz. In case of fast standby exit, FIRC_DIV_SEL is 00b and FIRC_CLK is 24 MHz.
2. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] values are don't care since QSPI_SFCK and TRACE_CLK are not used in this use case.
3. MC_CGM.MUX_0_CSC[SELCTL] must equal 0.

23.7.2.4 Option D - Low-Speed RUN mode (CORE_CLK @ 48 MHz)

Table 130. Option D - Low-Speed RUN mode (CORE_CLK @ 48 MHz)

Clocking options (SYS_CLK = FIRC_CLK)	Clock frequencies
	MWCT2xxxS
PLL VCO frequency	—

Table continues on the next page...

Table 130. Option D - Low-Speed RUN mode (CORE_CLK @ 48 MHz) (continued)

Clocking options (SYS_CLK = FIRC_CLK)	Clock frequencies
	MWCT2xxxS
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	—
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	48 MHz (11b)
PLL_PHI1_CLK-related clocks [1]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	—
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	—
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	—
PLL_PHI0_CLK-related clocks [2]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	—
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	—
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	—
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	48 MHz (0000b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	48 MHz (0000b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	24 MHz (0001b)

Table continues on the next page...

Table 130. Option D - Low-Speed RUN mode (CORE_CLK @ 48 MHz) (continued)

Clocking options (SYS_CLK = FIRC_CLK)	Clock frequencies
	MWCT2xxxS
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	48 MHz (0000b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	48 MHz (0000b)

1. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] values are don't care since QSPI_SFCK and TRACE_CLK are not used in this use case.
2. MC_CGM.MUX_0_CSC[SELCTL] must equal 0.

23.7.2.5 Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz)

Table 131. Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz)

Clocking options	Clock frequencies
	MWCT2xxxS
PLL VCO frequency	—
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	—
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	3 MHz (10b)
PLL_PHI1_CLK-related clocks [1]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	—
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	—
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	—
PLL_PHI0_CLK-related clocks [2]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	—
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	—
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	—

Table continues on the next page...

Table 131. Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz) (continued)

Clocking options	Clock frequencies
	MWCT2xxxS
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	3 MHz (0000b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	3 MHz (0000b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	1.5 MHz (0001b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	3 MHz (0000b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	3 MHz (0000b)

1. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] values are don't care since QSPI_SFCK and TRACE_CLK are not used in this use case.
2. MC_CGM.MUX_0_CSC[SELCTL] must equal 0.

NOTE

For FIRC_CLK frequency modes less than 24 MHz, safety modules like the CMU_F*x_n* must be disabled for safety applications, because safety applications are to run on the PLL clocks. The CMU_F*x_n* will cause erroneous FHH events if not disabled.

23.7.2.6 Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 kHz)

Table 132. Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 kHz)

Clocking options (SYS_CLK = (FIRC_CLK ÷ 8) ÷ 8)	Clock frequencies
	MWCT2xxxS
PLL VCO frequency	—
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	—
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	3 MHz (10b)
PLL_PHI1_CLK-related clocks [1]	

Table continues on the next page...

Table 132. Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 kHz) (continued)

Clocking options (SYS_CLK = (FIRC_CLK ÷ 8) ÷ 8)	Clock frequencies
	MWCT2xxxS
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	—
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	—
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	—
PLL_PHI0_CLK-related clocks [2]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	—
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	—
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	—
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	750 KHz (0011b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	750 KHz (0011b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	375 KHz (0111b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	750 KHz (0011b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	750 KHz (0011b)

1. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] values are don't care since QSPI_SFCK and TRACE_CLK are not used in this use case.
2. MC_CGM.MUX_0_CSC[SELCTL] must equal 0.

NOTE

For FIRC_CLK frequency modes less than 24 MHz, safety modules like the CMU_F*x_n* must be disabled for safety applications, because safety applications are to run on the PLL clocks. The CMU_F*x_n* will cause erroneous FHH events if not disabled.

23.7.2.7 Option F - Operation in 1:1 mode with CORE_CLK and AXBS_CLK at same speed

Table 133. Option F - Operation in 1:1 mode with CORE_CLK and AXBS_CLK at same speed

Clocking options	Clock frequencies		
	MWCT2D17S and MWCT2D16S		MWCT2015S and MWCT2016S
PLL VCO frequency	960 MHz		960 MHz
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	480 MHz (02h)		240 MHz (04h)
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	—		
PLL_PHI1_CLK-related clocks [1]			
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	240 MHz (0001b)	160 MHz (0010b)	48 MHz (0100b)
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	120 MHz (001b)	80 MHz (001b)	—
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	**For fast pads**		—
	120 MHz (001b)	80 MHz (001b)	—
	For medium pads		—
	48 MHz (100b)	40 MHz (011b)	—
PLL_PHI0_CLK-related clocks [2]			
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	160 MHz (010b)		80 MHz (010b)
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	80 MHz (001b)		80 MHz (000b)

Table continues on the next page...

Table 133. Option F - Operation in 1:1 mode with CORE_CLK and AXBS_CLK at same speed (continued)

Clocking options	Clock frequencies	
	MWCT2D17S and MWCT2D16S	MWCT2015S and MWCT2016S
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	160 MHz (000b)	—
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	80 MHz (001b)	80 MHz (000b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	40 MHz (011b)	40 MHz (001b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	40 MHz (011b)	40 MHz (001b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	80 MHz (001b)	80 MHz (000b)
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	40 MHz (011b)	40 MHz (001b)

1. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1001b.
2. MC_CGM.MUX_0_CSC[SELCTL] must equal 1000b.

23.7.3 Gasket configurations in various clocking modes

Table 134. Gasket configurations in various clocking modes (for MWCT2D17S and MWCT2D16S)

Gasket configurations	Option A[1]	Option B[2]	Option C[3]	Option D[4]	Option E[5]	Option E2[6]	Option F[7]
eDMA (S0)	1:1	Bypass					
eDMA (S1)	Bypass						
HSE_B	1:2	1:1, 1:2	Bypass				
AIPS1/AIPS2	2:1		Bypass				
QuadSPI	2:1		Bypass				
PRAM/SRAM	WS enabled		WS disabled				
EMAC 32:64	1:1						
BDRAM 64:32	1:1						

1. See Option A - High Performance mode (CORE_CLK @ 160 MHz) for details.
2. See Option B - Reduced Speed mode (CORE_CLK @ 120 MHz) for details.
3. See Option C - Boot Standby mode (CORE_CLK @ 24 MHz) for details.
4. See Option D - Low-Speed RUN mode (CORE_CLK @ 48 MHz) for details.
5. See Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz) for details.
6. See Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 kHz) for details.
7. See Option F - Operation in 1:1 mode with CORE_CLK and AXBS_CLK at same speed for details.

Table 135. Gasket configurations in various clocking modes (for MWCT2015S and MWCT2016S)

Gasket configurations	Option B[1]	Option C[2]	Option D[3]	Option E[4]	Option E2[5]	Option F[6]
HSE_B	1:1, 1:2	Bypass				
AIPS1	2:1	Bypass				
PRAM/SRAM	WS disabled					
BDRAM 64:32 (TCM WS)	1:1					

1. See Option B - Reduced Speed mode (CORE_CLK @ 120 MHz) for details.
2. See Option C - Boot Standby mode (CORE_CLK @ 24 MHz) for details.
3. See Option D - Low-Speed RUN mode (CORE_CLK @ 48 MHz) for details.
4. See Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz) for details.
5. See Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 kHz) for details.
6. See Option F - Operation in 1:1 mode with CORE_CLK and AXBS_CLK at same speed for details.

23.7.4 Default clock configuration

At reset recovery, the chip runs on the FIRC_CLK as the default configuration as shown in Option C - Boot Standby mode (CORE_CLK @ 24 MHz). Clocking configuration Option_C is the default configuration out of reset with the HSE_B core as the boot core. The Cortex-M7_*n* application core clocks are gated by default. You need to enable the core clocks by the configuring the corresponding core clock enable bits shown in Core Clock Gating.

23.7.5 PCFS

The chip supports software-controllable PCFS for MC_CGM MUX_0 (see section "Progressive Clock Frequency Switching (PCFS)" in the "Clock Generation Module (MC_CGM)" chapter for details). PCFS increases and decreases the frequency in steps, avoiding any overshoots or undershoots. When a functional reset event occurs with PCFS enabled, the PCFS process runs and is then followed by the divider configuration updates.

23.7.6 Updating dividers: crossbar halt handshake

The clock divider update process consists of the crossbar halt handshake sequence (see section "Clock dividers update" in the "Clock Generation Module (MC_CGM)" chapter for the clock divider update process). A divider update asserts a request to the crossbar switch to halt any transaction that is in process. The dividers are updated when the crossbar switch acknowledges the request for halt. The halt request disables the crossbar switch gaskets in the following order:

1. Core gaskets (HSE_B gaskets)
2. Crossbar switch (AXBS)
3. Flash AXBS bridge
4. PRAM/SRAM gasket

Dividers are updated after the gaskets acknowledge the halt request.

23.7.7 Changing system clock configurations

Software sequence for switching clock configurations from FIRC to PLL or PLL to PLL:

1. Before changing the system clock dividers or before system clock switching, the communication modules working on the system clock should be disabled by the software to avoid any erroneous communication during the clock transition.
2. The peripherals working on the system clock should be clock-gated used the MC_ME.PRTNx_COFBy_CLKEN configurations. The peripherals working on non-system clocks which are not being switched, can continue to operate.
3. All cores must be clock-gated, except the core being used to control the clock switching.

4. MC_CGM divider configurations can be done to change divider values as described above. The requisite system divider frequency relations must be ensured.
5. Configure MC_CGM_MUX_0_DIV_TRIG_CTRL (enable HHEN and TCTL).
6. Configure MC_CGM_MUX_0_DIV_TRIG.
7. System clock switching can be done by configuration of MC_CGM_MUX_0_CSC[SELCTL]

Software sequence for switching clock configurations from PLL to FIRC clock switching:

1. Before changing the system clock dividers or before system clock switching, the communication modules working on the system clock should be disabled by the software to avoid any erroneous communication during the clock transition.
2. The peripherals working on the system clock should be clock-gated used the MC_ME.PRTNx_COFBy_CLKEN configurations. The peripherals working on non-system clocks which are not being switched, can continue to operate.
3. All cores must be clock-gated, except the core being used to control the clock switching.
4. System clock switching can be done by configuration of MC_CGM_MUX_0_CSC[SELCTL].
5. MC_CGM divider configurations can be done to change divider values as described above. The requisite system divider frequency relations must be ensured.
6. Configure MC_CGM_MUX_0_DIV_TRIG_CTRL (enable HHEN and TCTL).
7. Configure MC_CGM_MUX_0_DIV_TRIG.

NOTE

When enabling PLL, the PMC last mile regulator should be enabled first by configuring PMC_CONFIG[LMEN] and PMC_CONFIG[LMBCTLEN] (if using an external BJT). The last mile regulator must be disabled after PLL is disabled.

23.8 Clock monitoring

The chip contains an independent clock monitoring mechanism which signals malfunctions in the clocking system. This chip consists of six Clock Monitoring Units (CMU_F*x*_[0:6]) for monitoring system clock and clocking module outputs. Figure 103 and Figure 104 show a lower-level view of the clocking monitoring system. Table 136 describes each CMU instance. Each CMU instance provides an independent interrupt or reset indication when the clock signal is out of range or lost. The CMU_FM_*n* provides a timeout indication in case there is a loss of metered clock. Your software must periodically check the CMU_FM_1 and CMU_FM_2 status within the chip FTTI (as specified in the Safety Manual).

NOTE

You must disable the CMU corresponding to the system clocks if the application changes the system clock source or changes the system clock divider configuration.

You must disable the CMU monitoring a clock source before disabling the clock source, then enable it after enabling the clock source.

The CMUs should be turned ON only after device has moved to PLL source (wherein LMR is ON).

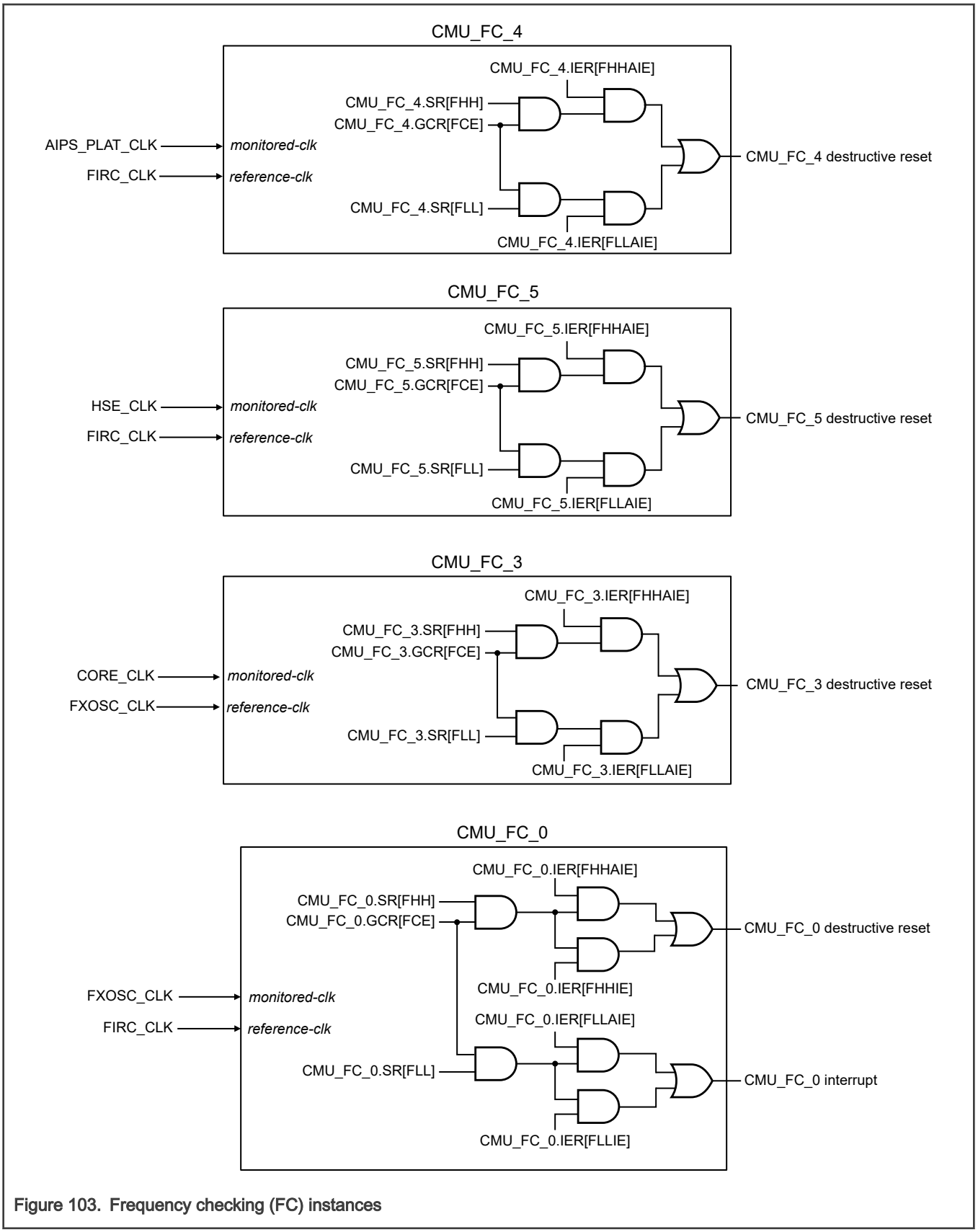


Figure 103. Frequency checking (FC) instances

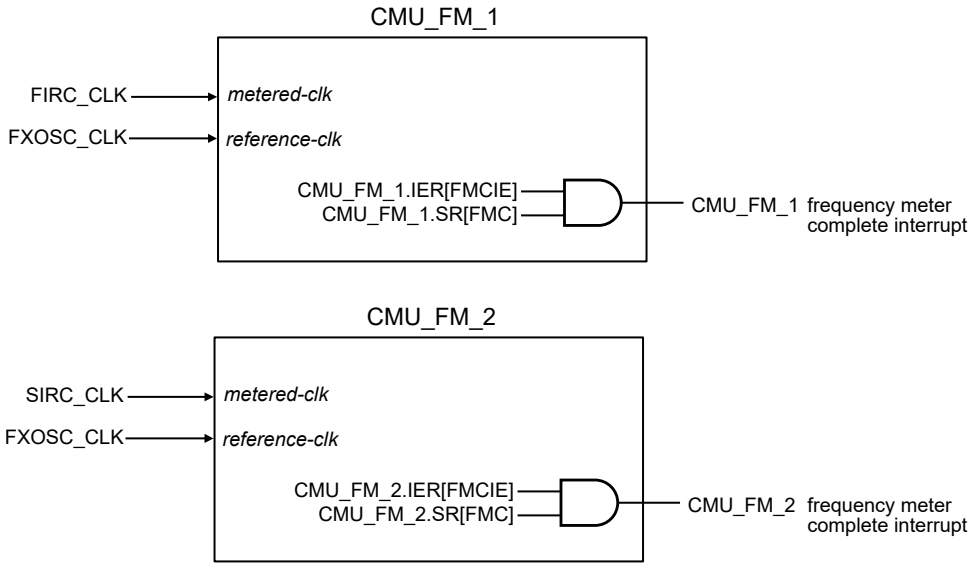


Figure 104. Frequency metering (FM) instances

Table 136. System clock monitors

CMU	Reference clock	Monitored or metered clock	Failure reaction	Monitoring type description
CMU_FC_0	FIRC_CLK	FXOSC_CLK	Destructive reset or interrupt	Precision over and under frequency
CMU_FM_1	FXOSC_CLK	FIRC_CLK	Interrupt	Current frequency measurement periodically triggered by software
CMU_FM_2	FXOSC_CLK	SIRC_CLK	Interrupt	Current frequency measurement periodically triggered by software
CMU_FC_3	FXOSC_CLK	CORE_CLK	Destructive reset	Precision over and under frequency
CMU_FC_4	FIRC_CLK	AIPS_PLAT_CLK	Destructive reset	Precision over and under frequency
CMU_FC_5	FIRC_CLK	HSE_CLK	Destructive reset	Precision over and under frequency

23.9 Glossary

MODULE_CLK	Module operating clock
REG_INTF_CLK	Module register interface clock used for register read and write
PCFS	Progressive Clock Frequency Switching (see section PCFS for details)
POR	Power On Reset
SBC	System Basis Chip (see NXP SBC portfolio)
FLL	Frequency lower than low frequency reference
FHH	Frequency higher than high frequency reference

FTTI	Fault Tolerance Time Interval

Chapter 24
Clock Generation Module (MC_CGM)

24.1 Chip-specific MC_CGM information

24.1.1 Associated content references

See the Clocking chapter for details pertaining to these:

- Chip clocking
- MC_CGM clock source mapping (see the "MC_CGM clock sources mapping" section for this)
- MC_CGM clock multiplexers (see the "MC_CGM clock multiplexers" section for this)

CAUTION

MC_CGM clock multiplexer configurations to non-supported and reserved clock sources are prohibited and can result in chip malfunctioning.

24.2 Introduction

The clock generation module (MC_CGM) is used to set up the configurable clock domains used by various chip blocks as per the application needs. It includes the clock multiplexers that allow software to select the desired clock sources for these domains. This is managed by the MC_CGM to ensure that the changing of the clock selection from one source to another occurs in a glitch-less fashion. In addition, the MC_CGM includes the clock dividers that can be configured by software.

See following figure for the MC_CGM block diagram:

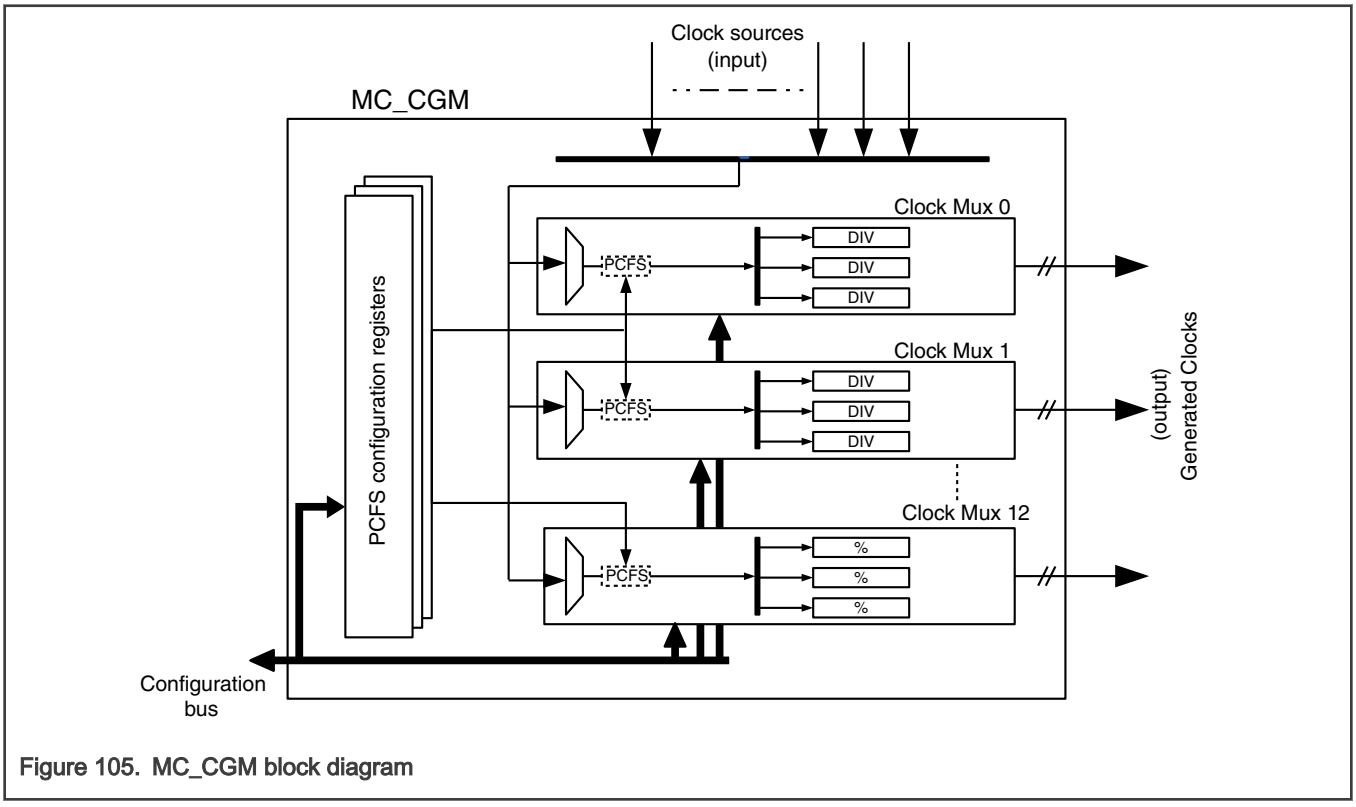


Figure 105. MC_CGM block diagram

NOTE

The block diagram is generic and does not necessarily reflect any specific MC_CGM implementation.

24.3 Features

MC_CGM includes the following features:

- Implements software configurable clock multiplexers for selecting from various clock sources
- Provides hardware-controlled multiplexers that guarantee glitchless transition, while the software-controlled multiplexers need a software sequence to ensure such a transition
- Provides software configurable automatic PCFS on certain clocks to minimize the impact of a sudden power consumption change through a gentle ramp-down and -up of the clock frequency when switching clock sources
- Implements software configurable clock dividers

24.4 Functional description

24.4.1 Clock selection multiplexer

Each of the clock multiplexers inside the MC_CGM either implements a fully hardware-controlled clock multiplexer or a software-controlled clock multiplexer. The following sections describe the two variants of the clock multiplexer.

24.4.1.1 Hardware-controlled clock multiplexer

In hardware-based clock multiplexing, the underlying assumption is that under some conditions, error or reset states, software may not be active. Therefore, the clock switching is fully hardware based and is glitchless. To facilitate clock switching requests with software, the MUX_n_CSC and MUX_n_CSS registers implement request and status for the clock multiplexer, the rest is managed in hardware. Using these registers, the software can monitor the state of the hardware-based clock multiplexer and also make clock switching requests. It is recommended that a new clock switch request is given only when there are no pending/ongoing clock switching requests. However, a switch to the safe clock, that is, FIRC, can be performed at any instance of time. A switch to the safe clock is always completed. Software should ensure that while making a software-based switch to safe clock, the register configuration clock should be available, at least for completing the write to the MUX_n_CSC register. This means that the clock should be running for the register write to complete. Hardware clock multiplexer also supports hardware-based switch to safe clock, which is requested externally to MC_CGM (for example, by MC_RGM). For a hardware-based switch to safe clock, it is not required to have a register configuration clock for MC_CGM.

NOTE

- "Switch to safe clock" from software is requested by programming the MUX_n_CSC[SAFE_SW] bit field only and not by combining the MUX_n_CSC[CLK_SW] and MUX_n_CSC[SELCTL] bit fields of the MUX_n_CSC register. Writes to other fields are ignored when requesting switch for safe clock.
- After the switch to the safe clock requested by the MC_RGM has completed, the MC_RGM also requests the clock dividers to switch to their default values. This hardware-triggered divider update can be monitored in the same manner as for software-configured updates, and software should use it to ensure that the configuration update has completed before reconfiguring the clocks.
- Write accesses to the MUX_n_CSC register with clock select pointing to the "reserved" input clock source are aborted with bus transfer errors.

See Figure 106 that shows the flowchart representation for the sequence of steps to be followed when making a clock switch request to hardware-controlled clock multiplexer. Switch to safe clock can be requested at any instance of time, and for clarity reasons, it is not shown in Figure 106.

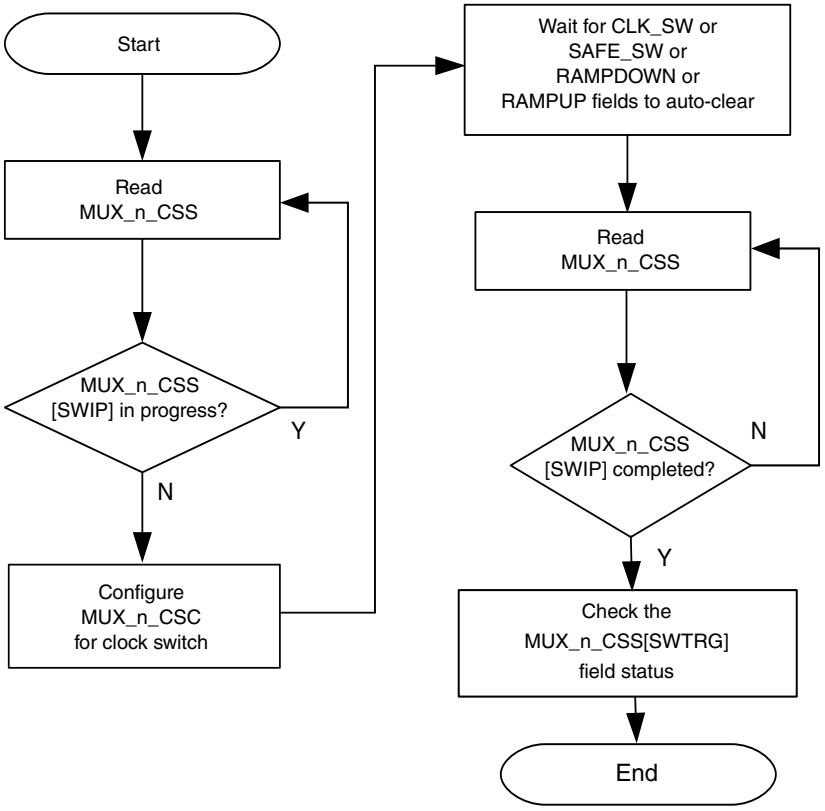


Figure 106. Flow for clock switch request on hardware-controlled clock multiplexer

NOTE

- A switch to the safe clock command always leads to a ramp-down from the currently selected clock and then a switch to "safe clock", except when there is an ongoing clock switch requested by the software without ramp-up and ramp-down. A safe clock switch request when there is an ongoing clock switch without ramp-up or ramp-down results in a switch to the safe clock without performing a ramp-down (either by MC_RGM or provided using the MUX_n_CSC register) does not perform a ramp-down before switching to "safe clock".
- The above flowchart steps can be preceded by points 1 and 3 below:
  1. Disable the divider.
  2. Switch clocks through hardware multiplexer.
  3. Enable and configure the dividers (atomic write instruction).

24.4.1.2 Software-controlled clock multiplexer

In software-based clock multiplexing, the underlying assumption is that the software is always available and there are no error or reset conditions in the chip. This implies that a glitchless switch between input clock sources of MC_CGM MUX can be achieved by following a sequence of steps in software. The software-based clock multiplexer implements a clock gate at the output of the clock mux. The software can gate the selected clock of MC_CGM MUX using a synchronous/graceful clock gate bit (that is, MUX_n_CSC[CG]) or a forced clock gate bit (that is, MUX_n_CSC[FCG]). The hardware does not guarantee that any glitches will escape when using forced clock gating. When a forced clock gate bit is written to 1, the internal clock gate forcefully gates the selected clock to logic-0, therefore, to avoid clock glitches, it should be ensured that the selected clock source is not running. See Figure 107 that shows the flowchart representation for the sequence of steps to be followed when making a clock switch request to software-controlled clock multiplexer. No switch to safe clock is supported in software-controlled clock multiplexer.

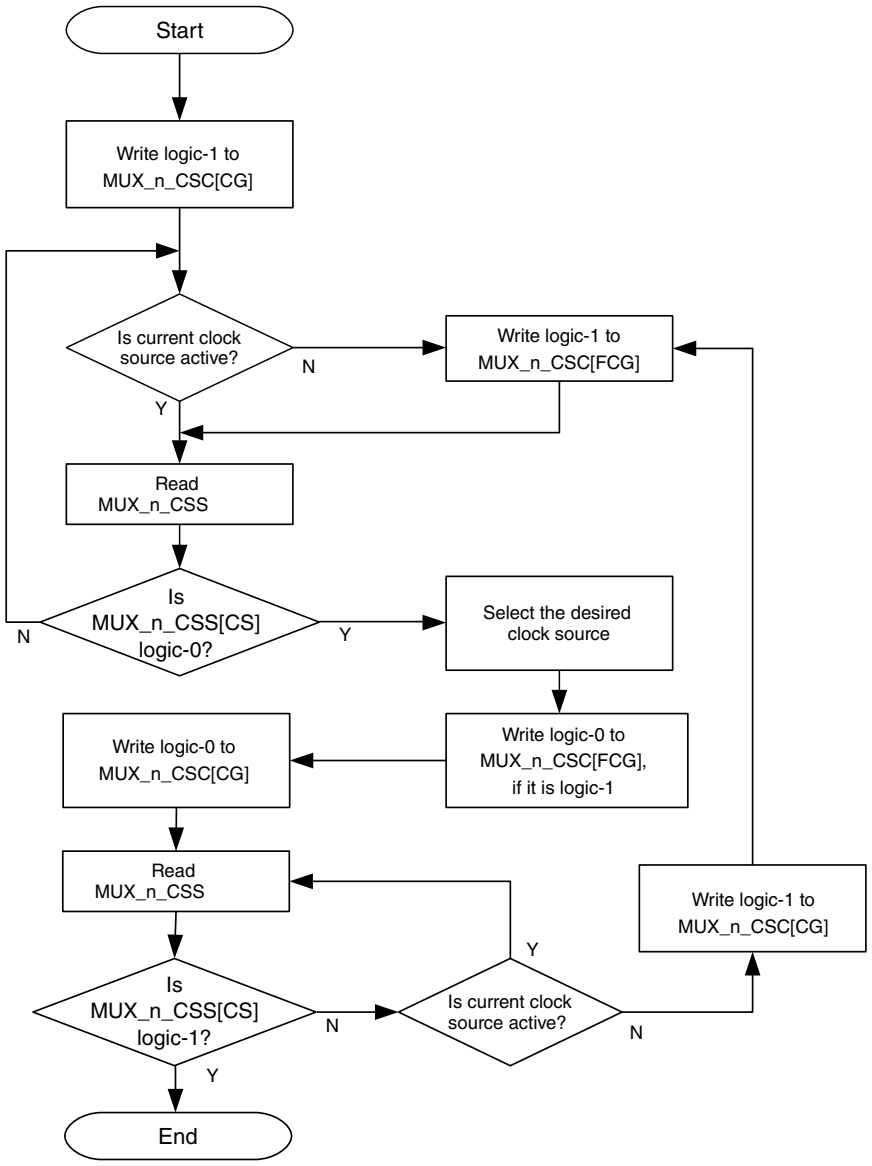


Figure 107. Flow for clock switch request on software controlled clock multiplexer

NOTE

- Ensure that before using the force gate bit, any IP or other logic using the clock of MC_CGM MUX is in the inactive state or a clock glitch resulting from usage of forced gating does not effect the IP (that is, it is clock gated after the MC_CGM).
- In Figure 107, the clock source to be selected should be active at the time of clock switch, else the MUX_n_CSS[CS] field will remain set to logic-0. In case the clock source to be selected becomes inactive (that is, loss of clock, and so on) during the switching operation, the switch to another clock source can be initiated by writing both the MUX_n_CSC[CG] and MUX_n_CSC[FCG] fields to logic-1.
- Writing a 'reserved' value for the MUX_n_CSC[SELCTL] field may result in an unpredictable clock at the output of the clock multiplexer.

NOTE

The above flowchart steps can be preceded by points 1 and 3 below:

1. Disable the divider.
2. Switch clocks through software multiplexer.
3. Enable and configure the dividers (atomic write instruction).

24.4.2 PCFS

MC_CGM implements PCFS when changing clock source at an MC_CGM clock mux. The PCFS is only implemented in a hardware-controlled clock multiplexer and not in a software-controlled clock multiplexer. It allows a gradual load change for a power/voltage supply unit by employing a gradual frequency changeover from one to another. The frequency changeover is achieved by clock division of input clock source with a sequence of division values, where frequency ramp down and ramp up is achieved when the sequence of division values are ascending and descending in nature, respectively. As the clock division (fractional) is implemented in digital logic, therefore, it is a coarse-level clock division rather than being an accurate-level division, implying that the duty cycle of the progressively divided clock can vary with time.

The PCFS feature is utilized when a drain current to frequency relationship is known, that is, for a given drain current what is the maximum allowed change in frequency (f_{chg}). The f_{chg} is the first input parameter for PCFS and other parameters for PCFS are specified or calculated in relation to FIRC.

The PCFS hardware generates the clock division factors based on certain values that are programmed into the PCFS configuration register. The following pseudo codes represent the generation of clock division value sequence (d_i).

```
/* ramp down division values (d_n) with k steps*/
      delta_1 = RATE/1000;
      delta_2 = RATE/1000;
      d_0     = 1.0;
      for i=1 to k-1 do
        d_i = d_i-1 + delta_1;
        delta_1 = delta_1 + delta_2;
      endfor
```

```
/* ramp up division values (d_n) with k steps*/
      delta_1 = RATE*k/1000;
      delta_2 = RATE/1000;
      d_0     = 1.0 + RATE*k*(k+1)/2;
      for i=1 to k-1 do
        d_i = d_i-1 - delta_1;
        delta_1 = delta_1 - delta_2;
    endfor
```

As the generation of clock division values is not a closed-bounded function, calculating RATE for a given f_{chg} is an iterative process. Find a value of RATE that produces a clock division sequence, which when used does not lead to a frequency change greater than f_{chg}. See Table 137 that tabulates some of the RATE values against a_{max}, where

$$a_{max}=f_{chg}/F_i$$

where, F_i is the frequency of i^{th} input clock source of the clock mux.

Table 137. PCFS RATE values

a_{max}	PCFS rate
0.005	12
0.01	48

Table continues on the next page...

Table 137. PCFS RATE values (continued)

a_{max}	PCFS rate
0.15	112
0.20	184

The last clock division factor in case of ramp-down or the first clock in case of ramp-up (following a ramp-down procedure), should be such that clock switch from any input clock source to another should be termed as safe, indicating that load changes have sustainable power effects. This frequency level is referred to as "safe frequency", equivalent to frequency of FIRC referred to as "safe clock" with frequency "safe clock frequency" (f_{safe}). The last clock division factor in the sequence of clock division factors happens after "k" steps. The factor k (=steps) is calculated by using the following formula:

```
k = ceil(0.5 + sqrt(0.25 - (2000 * (1 -(Fi/fsafe)) / RATE)))
```

Using the above formula, all the PCFS register configuration values can be calculated for a given frequency of a clock source:

```
PCFS_DIVEi.DIV = (Fi/fsafe)*1000-1;
        PCFS_DIVCi.INIT = RATE * k;
        PCFS_DIVCi.RATE = RATE;
        PCFS_DIVSi.DIV  = 999 + (RATE * k * (k+1)/2);
```

See Figure 108 that shows a graphical representation of the change in frequency, which happens during PCFS ramp-down and ramp-up.

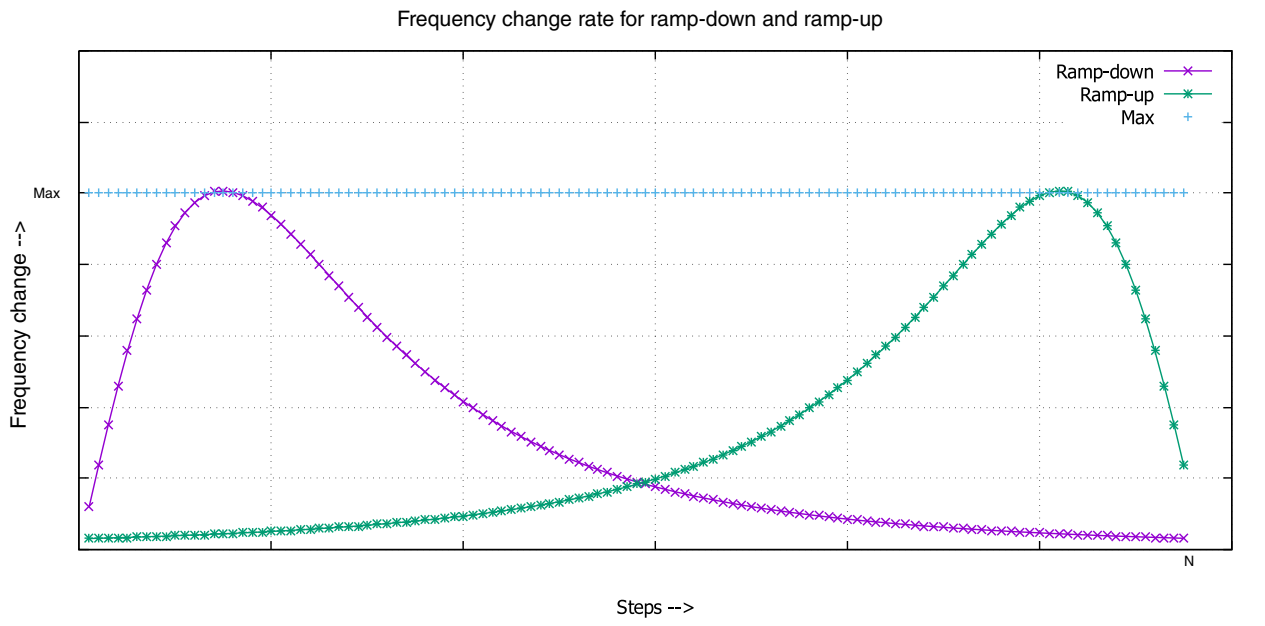


Figure 108. PCFS steps vs frequency changes

For any given clock source, if its frequency is less than that of FIRC, then its corresponding registers should be programmed to default values, where the default values are such that PCFS divider start and PCFS divider end values are same and equal to divide-by-1. The default values ensure that no progressive clock division is done when a clock switch request is given to switch from or to that source.

NOTE

Calculate the minimum frequency during the PCFS RAMPDOWN and RAMPUP operations by using the formula:

```
(FIRC/((PCFS_DIVE+1)/1000)) MHz
```

24.4.2.1 PCFS control

The PCFS operation is configured by a set of configuration registers. One set pertains to the calculation of the clock division factors, which are PCFS_DIVC, PCFS_DIVE, PCFS_DIVS, and PCFS_SDUR, while the registers MUX_n_CSC and MUX_n_CSS implement trigger and status of the PCFS operation. The clock division factors are expected to be programmed before any other MC_CGM operation is initiated and remain unchanged. All the registers corresponding to the clock division factors should be programmed with FIRC as the configuration clock and before doing any clock switch or PCFS operation on any of the MC_CGM mux. It should be noted that the default values of the register corresponding to clock division factors are such that only the clock division factor is calculated by hardware that is divide-by-1. The PCFS operation is always triggered when the safe clock request is generated except when there is an on-going clock switch without ramp-up or ramp-down. Therefore, the software needs to ensure that the PCFS configuration is complete and correct.

While configuring MUX_n_CSC, only valid PCFS and clock switch requests should be provided. PCFS or clock switch requests should only be provided if the PCFS operation is in the idle state. If there is an ongoing PCFS operation, it is recommended not to provide any new PCFS triggers (except switch to safe clock) until the ongoing operation is completed. Switch to safe clock via hardware or by register configuration can be provided at any instance of time and is always completed. Valid combinations of PCFS and clock switches triggers are listed in Table 138. All the PCFS commands should be atomic in nature, which means a single register write should provide complete PCFS sequence to be executed that is ramp-down, clock switch, and ramp-up.

Table 138. Valid PCFS and clock switch requests

PCFS operation state	Command
Idle	Ramp-down, clock switch, and ramp-up
Idle	Clock switch only (without ramp-up or ramp-down)

When a switch to safe clock is provided by writing to MUX_n_CSC, then writes to other register fields are ignored.

24.4.2.2 Clock source powerup and selection

This section provides guidelines for powering up a given clock source and selecting it at the MC_CGM clock multiplexer.

Following is the powerup procedure for a clock source:

1. Configure the parameter, if any.
2. Configure the powerup (or powerdown) control field.
3. Wait for the powerup status indication.

After a powerup indication, the clock source can be selected to provide output clock at an MC_CGM clock multiplexer. A clock-monitoring setup can also be activated on the powered-up clock source.

See Figure 109 that shows a flow chart representation of this sequence.

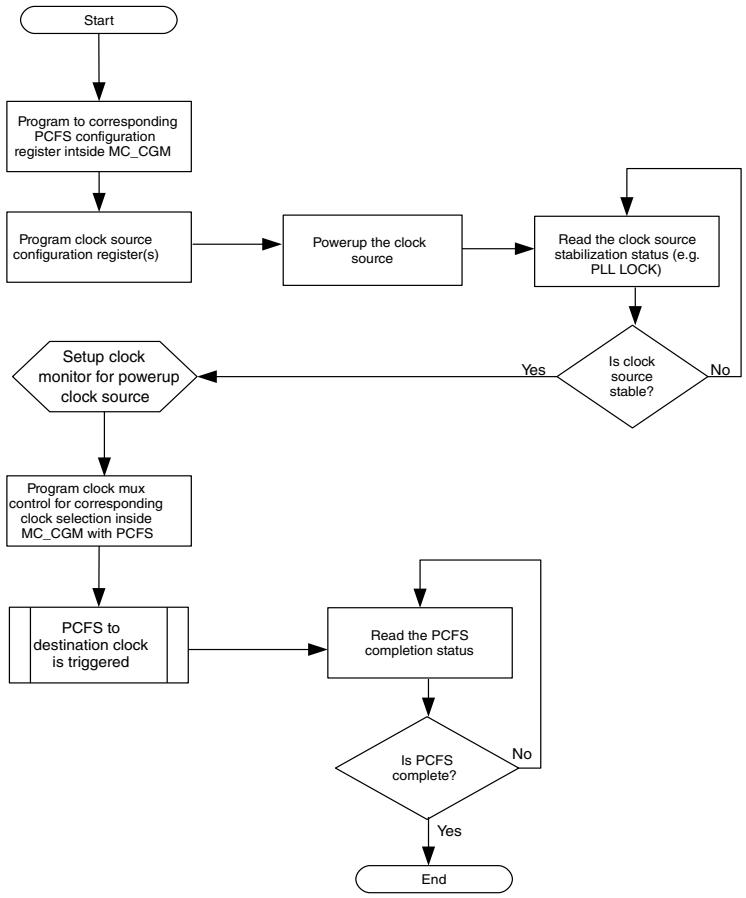


Figure 109. Clock source powerup and selection procedure

24.4.2.3 Clock source powerdown and deselection

This section provides guidelines for powering down a given clock source and deselecting it at MC_CGM clock multiplexer.

Following is the powerdown procedure for a clock source:

1. Deselect the clock source at all MC_CGM clock multiplexers.
2. Configure the powerup (or powerdown) control field.
3. Wait for the powerdown status indication.

See Figure 110 that shows a flow chart representation of this sequence.

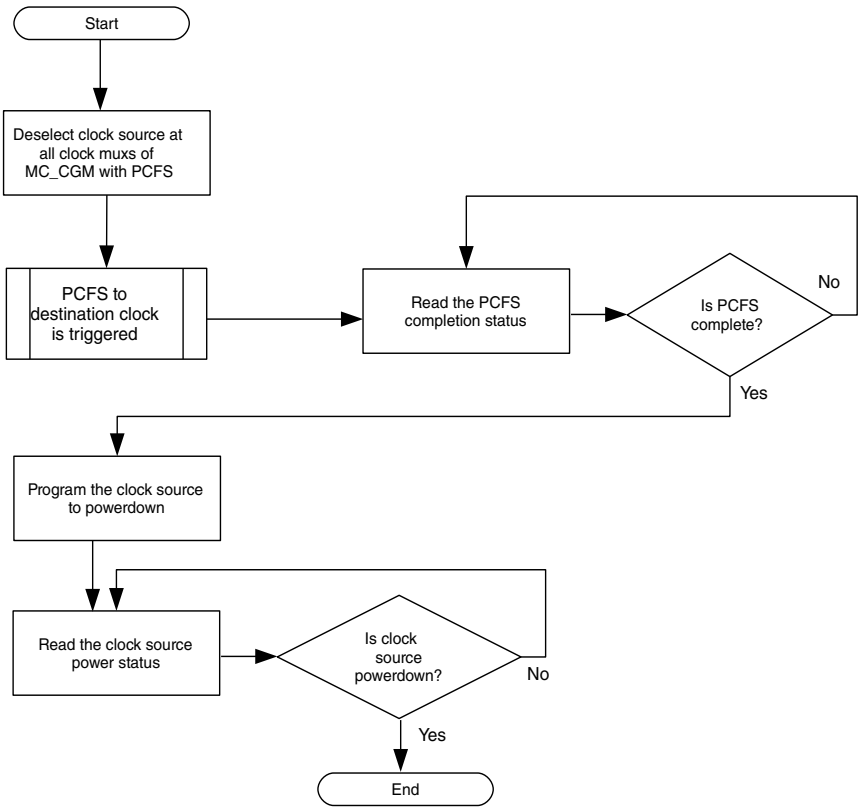


Figure 110. Clock source powerdown and clock source deselection procedure

24.4.2.4 Clock switch with load change

This section provides guidelines to switch between two high-frequency clock sources along with load changes in the system. Load change is referred to switching ON or OFF of logic/peripherals in the system, which has an effect of significant capacitance changes on the chip. This triggers the voltage regulation for the chip.

When a large number of peripherals or digital logic is enabled or disabled, it is recommended that this step should be performed at a low frequency. Independent of whether a clock switch is required, this criteria needs to be met. When a clock switch is required at two high frequencies, the recommended sequence is as follows:

1. Intermediately switch to FIRC.
2. Change load (that is, enable/disable peripherals).
3. Switch to the target clock.

See the following figure that shows a flow chart representation of this sequence.

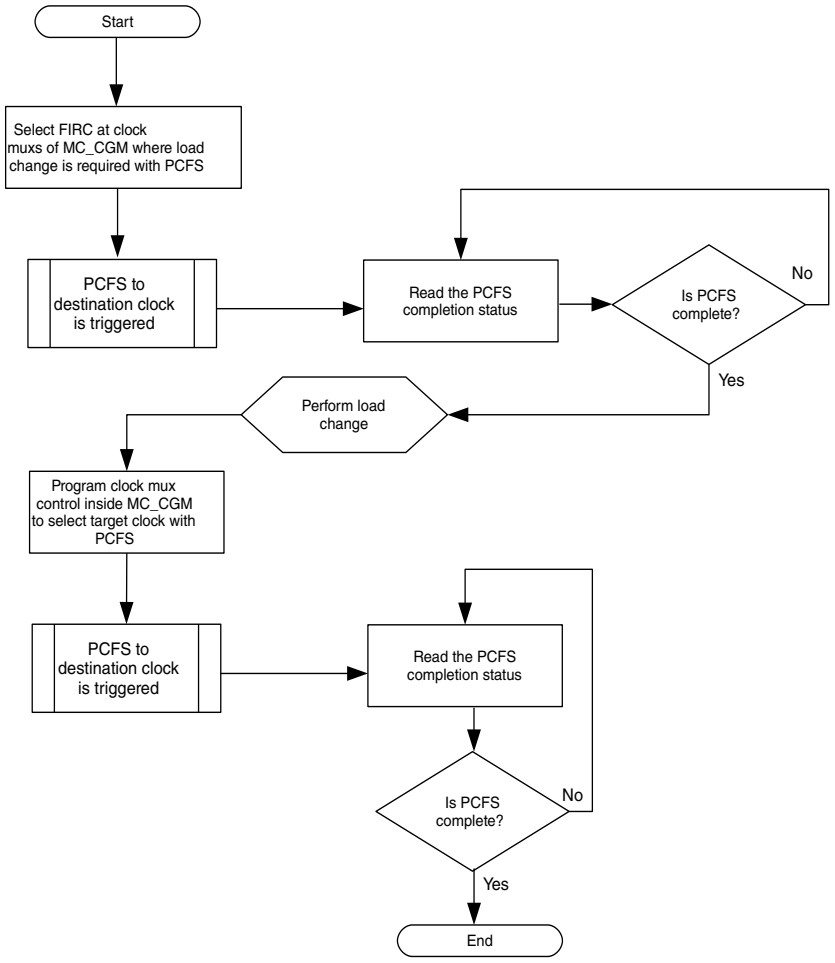


Figure 111. Clock switch with load change procedure

24.4.3 Clock dividers

Clock dividers are used for the generation of a divided clock that is used for running IPs or peripherals. The MC_CGM provides the following built-in dividers at each clock mux:

- 50% clock dividers

Each divider can be controlled by the Divider Enable (DE) bit and the Division Value (DIV) field. If a divider has its DE bit set to logic-0 in the respective configuration register, then that divider is disabled and the output divided clock is held to logic-0. If the DE bit is logic-1, the divider is enabled and provides a divided clock according to the value set in the DIV field.

24.4.3.1 50% clock divider

50% duty cycle dividers generate a real divided clock. The division factor is always an integer but is not restricted to even numbers. The rising edge of the divided clock is always synchronous to the rising edge of the divider source clock, but the falling edge is synchronous to the rising edge or the falling edge depending on whether the division factor is even or odd, respectively. If the input clock has a duty cycle of 50%, the divided output clock maintains the same 50% duty cycle. See Figure 112 that shows the 50% clock divider operation and its associated signals.

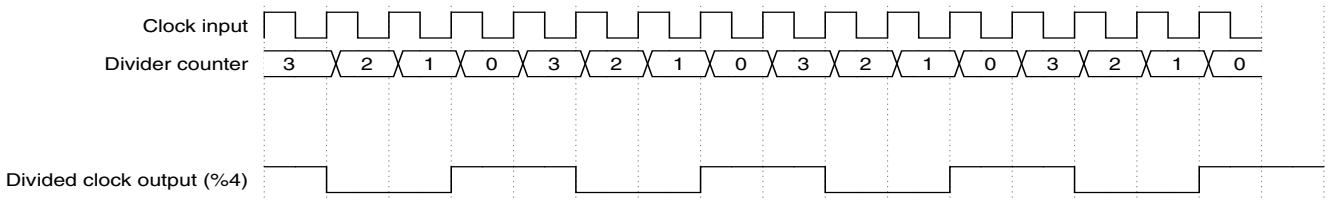


Figure 112. 50% clock divider programmed for divide by 4 (DIV=3)

A 50% divided clock can be considered asynchronous (not edge aligned) to other divided clocks from dividers at the same clock mux. 50% clock dividers are implemented without an active closed loop, and are expected not to get stuck if the input clock glitches for a single cycle.

24.4.3.2 Clock dividers update

To update the division value or the divider enable, the software should follow the procedure as shown in Figure 113. These updates happen only after the current division cycle has elapsed. However, if the phase of the clock divider is updated, the update happens independently from the state of the division cycle. Any update to the clock divider fields does not result in clock glitch either at the divided clock output or the phase-divided clock output.

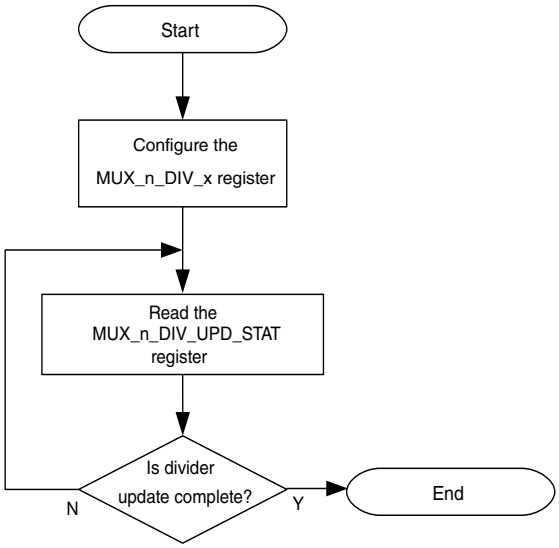


Figure 113. Flow for clock divider programming update

Following is the procedure for updating the dividers using the common trigger update:

1. Configure the MUX_n_DIV_TRIG_CTRL register.
2. Wait for the update to finish (until MUX_n_DIV_UPD_STAT is 0).
3. Update the clock dividers (only 50%) per the divider update procedure.
4. After the divider update is finished, perform a write operation on the MUX_n_DIV_TRIG register.
5. Wait for the update to finish, that is, until MUX_n_DIV_UPD_STAT is 0. During this period, the following process takes place:
   - Halt handshake is initiated if configured in step 1.
   - Clock dividers is updated only when AXBS is halted (that is, halt acknowledgment is received by MC_CGM). It is initiated, else the dividers are updated at alignment.
   - After the clock dividers are updated, MUX_n_DIV_UPD_STAT is asserted to 0.

- When the bit fields MUX_x_DIV_TRIG_CTRL[TCTL] and MUX_x_DIV_TRIG_CTRL[HHEN] are set to 1, then any write operation on trigger register will assert (MUX_n_DIV_UPD_STAT). Once the dividers are updated and aligned (MUX_n_DIV_UPD_STAT) will be deasserted.

NOTE

- The MUX_x_DIV_TRIG_CTRL[HHEN] bit should only be set when MUX_x_DIV_TRIG_CTRL[TCTL] is set, otherwise it may lead to misalignment of the dividers.
- In case of divider initialization by MC_RGM, a halt handshake protocol is initiated if the corresponding register bit is set and the clock dividers are initialized after the halt handshake protocol completion.

6. This completes the divider update.
7. When multiple writes to the dividers of same clock MUX is made without waiting for the previous update status signal to finish may lead to misalignment of the dividers.

For aligned dividers, the LCM of the division values programmed in the dividers of respective clock mux should be less than 100.

NOTE

Performing multiple writes to the divider without waiting for the earlier update to complete can lead to misalignment of the dividers.

Recommended software sequence for ensuring no undivided output at MC_CGM:

1. Reset is de-asserted
2. MC_RGM goes to IDLE
3. Enable the clock dividers of MC_CGM to provide FIRC clock so that reset of fixed dividers can be lifted.
4. Program the MC_CGM as per use case division values.
5. Switch the clock of MC_CGM Mux to desired one, and run the system.

24.5 MC_CGM register descriptions

MC_CGM implements a set of clock multiplexers that share PCFS configuration registers. MC_CGM registers have the following properties:

- All registers are 32-bit wide.
- Only 32-bit read and write accesses are supported.
- Read/write accesses of less than 32 bits terminate with an error.
- Writes to read-only register fields in writable registers are ignored and do not provide an error response.
- Writes to read-only registers are aborted with an error response.

24.5.1 MC_CGM memory map

MC_CGM base address: 402D_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PCFS Step Duration (PCFS_SDUR)	32	RW	0000_0000h
58h	PCFS Divider Change 8 Register (PCFS_DIVC8)	32	RW	0000_0000h
5Ch	PCFS Divider End 8 Register (PCFS_DIVE8)	32	RW	0000_03E7h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
60h	PCFS Divider Start 8 Register (PCFS_DIVS8)	32	RW	0000_03E7h
300h	Clock Mux 0 Select Control Register (MUX_0_CSC)	32	RW	0000_0000h
304h	Clock Mux 0 Select Status Register (MUX_0_CSS)	32	R	0008_0000h
308h	Clock Mux 0 Divider 0 Control Register (MUX_0_DC_0)	32	RW	8000_0000h
30Ch	Clock Mux 0 Divider 1 Control Register (MUX_0_DC_1)	32	RW	8000_0000h
310h	Clock Mux 0 Divider 2 Control Register (MUX_0_DC_2)	32	RW	8001_0000h
314h	Clock Mux 0 Divider 3 Control Register (MUX_0_DC_3)	32	RW	8000_0000h
318h	Clock Mux 0 Divider 4 Control Register (MUX_0_DC_4)	32	RW	8000_0000h
31Ch	Clock Mux 0 Divider 5 Control Register (MUX_0_DC_5)	32	RW	8003_0000h
320h	Clock Mux 0 Divider 6 Control Register (MUX_0_DC_6)	32	RW	8000_0000h
334h	Clock Mux 0 Divider Trigger Control Register (MUX_0_DIV_TRIG_CTRL)	32	RW	0000_0000h
338h	Clock Mux 0 Divider Trigger Register (MUX_0_DIV_TRIG)	32	W	0000_0000h
33Ch	Clock Mux 0 Divider Update Status Register (MUX_0_DIV_UPD_STAT)	32	R	0000_0000h
340h	Clock Mux 1 Select Control Register (MUX_1_CSC)	32	RW	0000_0000h
344h	Clock Mux 1 Select Status Register (MUX_1_CSS)	32	R	0008_0000h
348h	Clock Mux 1 Divider 0 Control Register (MUX_1_DC_0)	32	RW	0000_0000h
37Ch	Clock Mux 1 Divider Update Status Register (MUX_1_DIV_UPD_STAT)	32	R	0000_0000h
380h	Clock Mux 2 Select Control Register (MUX_2_CSC)	32	RW	0000_0000h
384h	Clock Mux 2 Select Status Register (MUX_2_CSS)	32	R	0008_0000h
388h	Clock Mux 2 Divider 0 Control Register (MUX_2_DC_0)	32	RW	0000_0000h
3BCh	Clock Mux 2 Divider Update Status Register (MUX_2_DIV_UPD_STAT)	32	R	0000_0000h
3C0h	Clock Mux 3 Select Control Register (MUX_3_CSC)	32	RW	0000_0000h
3C4h	Clock Mux 3 Select Status Register (MUX_3_CSS)	32	R	0008_0000h
3C8h	Clock Mux 3 Divider 0 Control Register (MUX_3_DC_0)	32	RW	0000_0000h
3FCh	Clock Mux 3 Divider Update Status Register (MUX_3_DIV_UPD_STAT)	32	R	0000_0000h
400h	Clock Mux 4 Select Control Register (MUX_4_CSC)	32	RW	0000_0000h
404h	Clock Mux 4 Select Status Register (MUX_4_CSS)	32	R	0008_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
408h	Clock Mux 4 Divider 0 Control Register (MUX_4_DC_0)	32	RW	0000_0000h
43Ch	Clock Mux 4 Divider Update Status Register (MUX_4_DIV_UPD_STAT)	32	R	0000_0000h
440h	Clock Mux 5 Select Control Register (MUX_5_CSC)	32	RW	0000_0000h
444h	Clock Mux 5 Select Status Register (MUX_5_CSS)	32	R	0002_0000h
448h	Clock Mux 5 Divider 0 Control Register (MUX_5_DC_0)	32	RW	8001_0000h
47Ch	Clock Mux 5 Divider Update Status Register (MUX_5_DIV_UPD_STAT)	32	R	0000_0000h
480h	Clock Mux 6 Select Control Register (MUX_6_CSC)	32	RW	0000_0000h
484h	Clock Mux 6 Select Status Register (MUX_6_CSS)	32	R	0002_0000h
488h	Clock Mux 6 Divider 0 Control Register (MUX_6_DC_0)	32	RW	8001_0000h
4BCh	Clock Mux 6 Divider Update Status Register (MUX_6_DIV_UPD_STAT)	32	R	0000_0000h
4C0h	Clock Mux 7 Select Control Register (MUX_7_CSC)	32	RW	0000_0000h
4C4h	Clock Mux 7 Select Status Register (MUX_7_CSS)	32	R	0008_0000h
4C8h	Clock Mux 7 Divider 0 Control Register (MUX_7_DC_0)	32	RW	0000_0000h
4FCh	Clock Mux 7 Divider Update Status Register (MUX_7_DIV_UPD_STAT)	32	R	0000_0000h
500h	Clock Mux 8 Select Control Register (MUX_8_CSC)	32	RW	0000_0000h
504h	Clock Mux 8 Select Status Register (MUX_8_CSS)	32	R	0008_0000h
508h	Clock Mux 8 Divider 0 Control Register (MUX_8_DC_0)	32	RW	0000_0000h
53Ch	Clock Mux 8 Divider Update Status Register (MUX_8_DIV_UPD_STAT)	32	R	0000_0000h
540h	Clock Mux 9 Select Control Register (MUX_9_CSC)	32	RW	0000_0000h
544h	Clock Mux 9 Select Status Register (MUX_9_CSS)	32	R	0008_0000h
548h	Clock Mux 9 Divider 0 Control Register (MUX_9_DC_0)	32	RW	0000_0000h
57Ch	Clock Mux 9 Divider Update Status Register (MUX_9_DIV_UPD_STAT)	32	R	0000_0000h
580h	Clock Mux 10 Select Control Register (MUX_10_CSC)	32	RW	0000_0000h
584h	Clock Mux 10 Select Status Register (MUX_10_CSS)	32	R	0008_0000h
588h	Clock Mux 10 Divider 0 Control Register (MUX_10_DC_0)	32	RW	0000_0000h
5BCh	Clock Mux 10 Divider Update Status Register (MUX_10_DIV_UPD_STAT)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5C0h	Clock Mux 11 Select Control Register (MUX_11_CSC)	32	RW	0000_0000h
5C4h	Clock Mux 11 Select Status Register (MUX_11_CSS)	32	R	0002_0000h
5C8h	Clock Mux 11 Divider 0 Control Register (MUX_11_DC_0)	32	RW	8000_0000h
5FCh	Clock Mux 11 Divider Update Status Register (MUX_11_DIV_UPD_STAT)	32	R	0000_0000h

24.5.2 PCFS Step Duration (PCFS_SDUR)

Offset

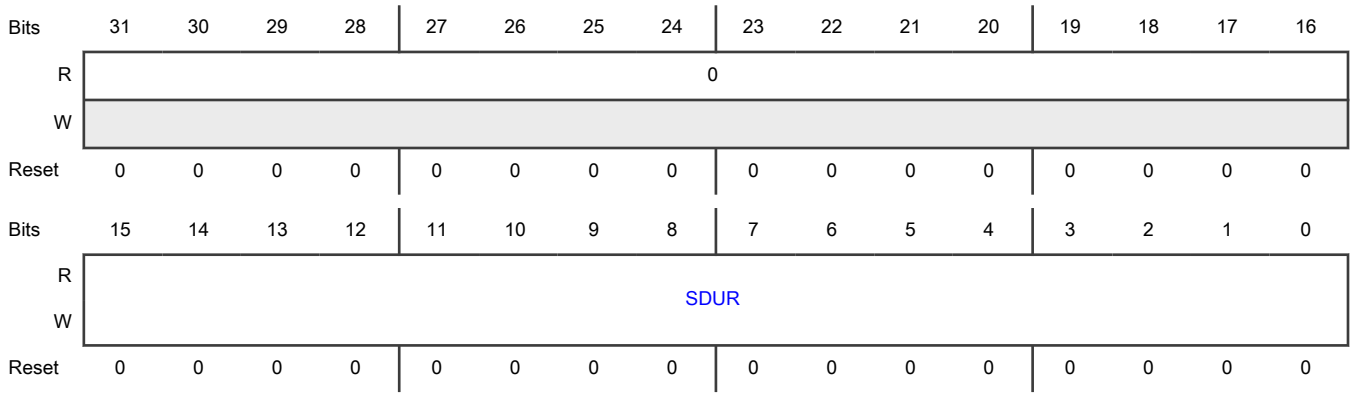
Register	Offset
PCFS_SDUR	0h

Function

This register specifies the step duration of each PCFS step. The value provided in this register specifies the PCFS step duration in terms of the number of cycles of FIRC.

This register is reset only by a destructive reset. For details, see PCFS.

Diagram



Fields

Field	Function
31-16 —	This field is reserved and reads return zeros.
15-0 SDUR	Step duration Count value of the step duration

24.5.3 PCFS Divider Change 8 Register (PCFS_DIVC8)

Offset

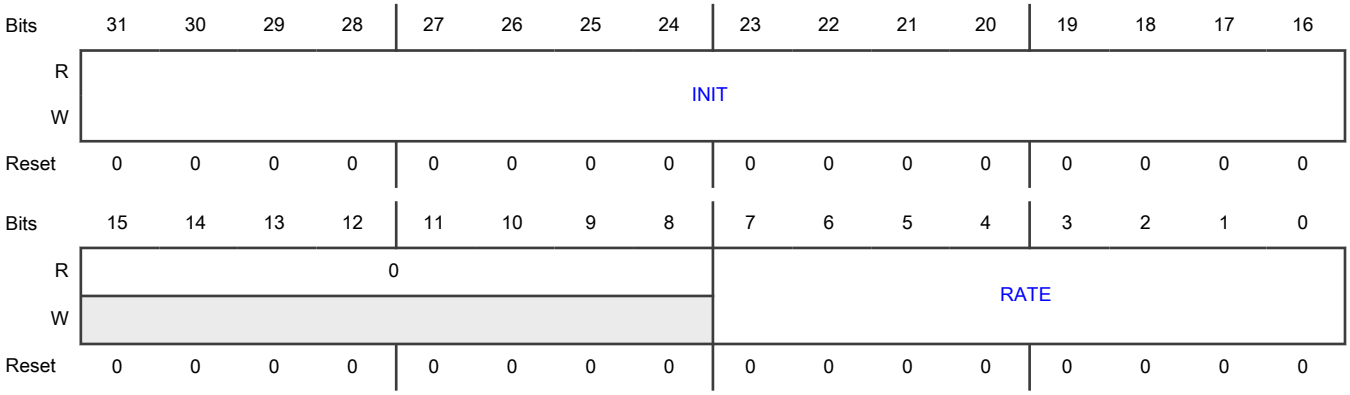
Register	Offset
PCFS_DIVC8	58h

Function

This register defines the rate of frequency change and initial change value on frequency ramp-up for the Progressive Clock Frequency switching of PLL_PHI0_CLK.

This register is reset only on destructive reset.

Diagram



Fields

Field	Function
31-16 INIT	Divider change initial value This field provides the initial change value of the clock divider for the clock ramp-up phase of PLL_PHI0_CLK.
15-8 —	This field is reserved and reads return zeros.
7-0 RATE	Divider change rate This value controls the change value of the clock divider for the clock ramp-up and ramp-down phase of PLL_PHI0_CLK.

24.5.4 PCFS Divider End 8 Register (PCFS_DIVE8)

Offset

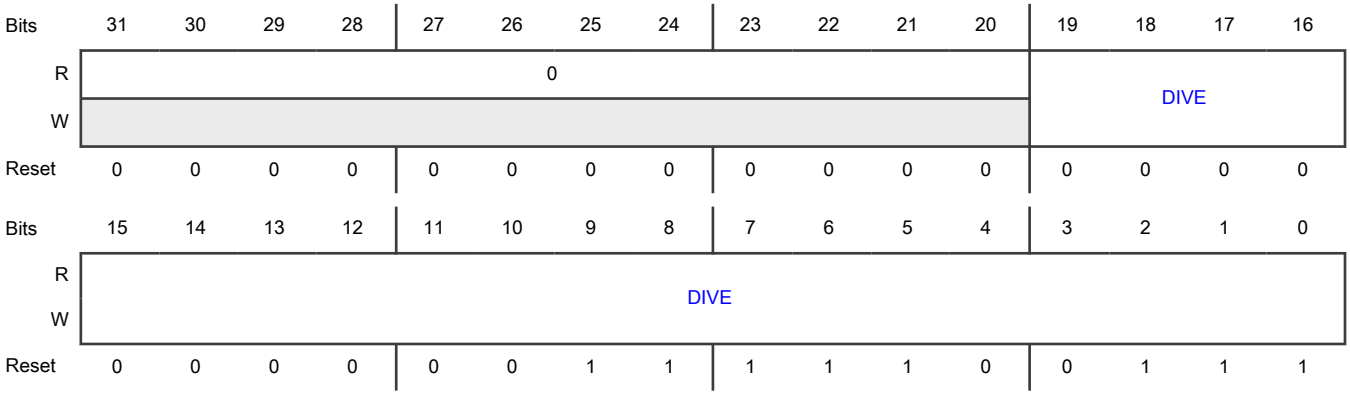
Register	Offset
PCFS_DIVE8	5Ch

Function

This register defines the final division value on frequency ramp-down for the progressive system clock switching of PLL_PHI0_CLK.

This registers is reset only on destructive reset.

Diagram



Fields

Field	Function
31-20 —	This field is reserved and reads return zeros.
19-0 DIVE	Divider end value This field provides the end value of the clock divider for the PLL_PHI0_CLK ramp-down phase.

24.5.5 PCFS Divider Start 8 Register (PCFS_DIVS8)

Offset

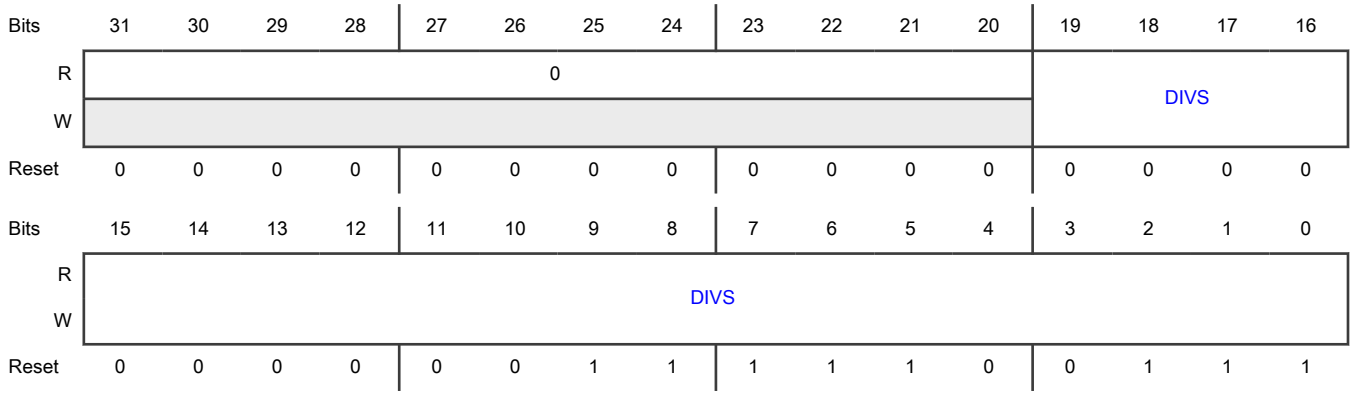
Register	Offset
PCFS_DIVS8	60h

Function

This register defines the initial division value on frequency ramp-up for the progressive system clock switching of PLL_PHI0_CLK.

This register is reset only on destructive reset.

Diagram



Fields

Field	Function
31-20 —	This field is reserved and reads return zeros.
19-0 DIVS	Divider start value This field provides the start value of the clock divider for the PLL_PHI0_CLK ramp-up phase

24.5.6 Clock Mux 0 Select Control Register (MUX_0_CSC)

Offset

Register	Offset
MUX_0_CSC	300h

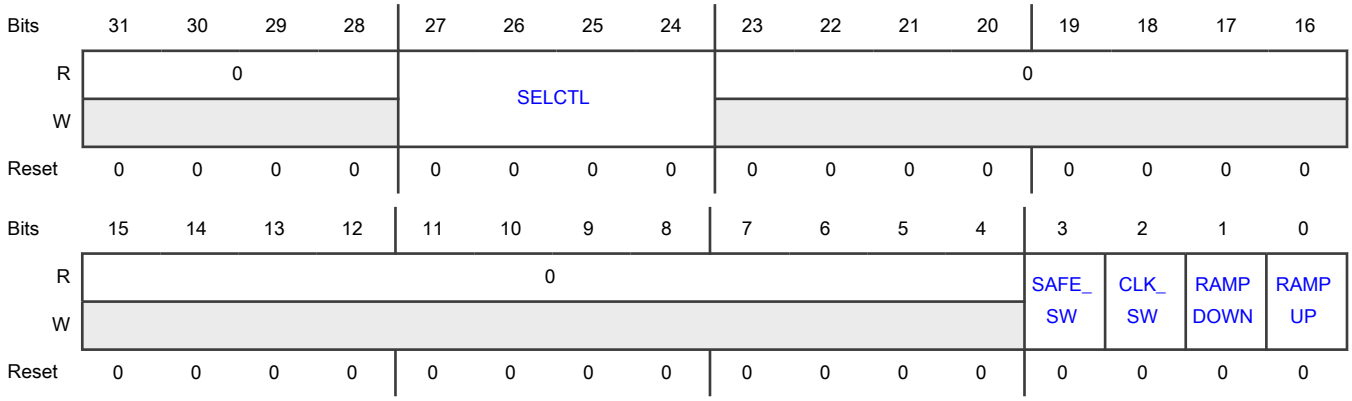
Function

This register provides the clock source selection control for clock mux 0. Clock mux 0 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

An update to all the PCFS-related fields of this register must be an atomic write, which means a single write must update the CLK_SW, RAMPDOWN, and RAMPUP fields. It is necessary to set both RAMPUP and RAMPDOWN bits together even if you want to trigger either RAMPUP or RAMPDOWN process otherwise the desired PCFS sequence will not be executed.

Diagram



Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELCTL	Clock source selection control Selects the source clock for clock mux 0. The reserved values are not displayed. 0000b - FIRC 1000b - PLL_PHI0_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 0. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1 RAMPDOWN	PCFS ramp-down Writing 1 to this bit makes a PCFS ramp-down request to clock mux 0. After a PCFS ramp-down operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
0 RAMPUP	PCFS ramp-up Writing 1 to this bit makes a PCFS ramp-up request to clock mux 0. After a PCFS ramp-up operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.

24.5.7 Clock Mux 0 Select Status Register (MUX_0_CSS)

Offset

Register	Offset
MUX_0_CSS	304h

Function

This register provides the current clock source selection status for clock mux 0.

This register is reset on destructive reset only.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	RAMP DOWN	RAMP UP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 0. The reserved values are not displayed. 0000b - FIRC 1000b - PLL_PHI0_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 0. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 0. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1 RAMPDOWN	PCFS ramp-down

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field provides an indication of whether a PCFS ramp-down operation was requested during the previous/ongoing request on clock mux 0. **NOTE** In case of safe clock switching, the ramp-down operation runs internally, but the value of the corresponding status field is not set to 1. 0b - No ramp-down operation was requested. 1b - Ramp-down operation was requested.
0 RAMPUP	PCFS ramp-up This field provides an indication of whether a PCFS ramp-up operation was requested during the previous/ ongoing request on clock mux 0. 0b - No ramp-up operation was requested. 1b - Ramp-up operation was requested.

24.5.8 Clock Mux 0 Divider 0 Control Register (MUX_0_DC_0)

Offset

Register	Offset
MUX_0_DC_0	308h

Function

This register controls the clock divider 0 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

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

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

Fields

Field	Function
31 DE	Divider enable 0b - Reserved 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.9 Clock Mux 0 Divider 1 Control Register (MUX_0_DC_1)

Offset

Register	Offset
MUX_0_DC_1	30Ch

Function

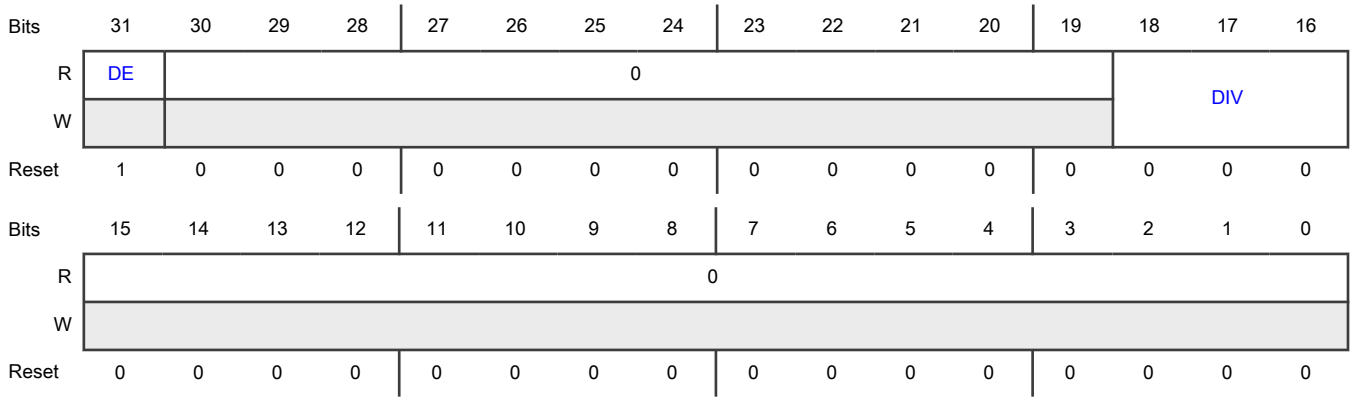
This register controls the clock divider 1 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Reserved 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.10 Clock Mux 0 Divider 2 Control Register (MUX_0_DC_2)

Offset

Register	Offset
MUX_0_DC_2	310h

Function

This register controls the clock divider 2 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

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

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

Fields

Field	Function
31 DE	Divider enable 0b - Reserved 1b - Divider is enabled.
30-20 —	This field is reserved and reads return zeros.
19-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.11 Clock Mux 0 Divider 3 Control Register (MUX_0_DC_3)

Offset

Register	Offset
MUX_0_DC_3	314h

Function

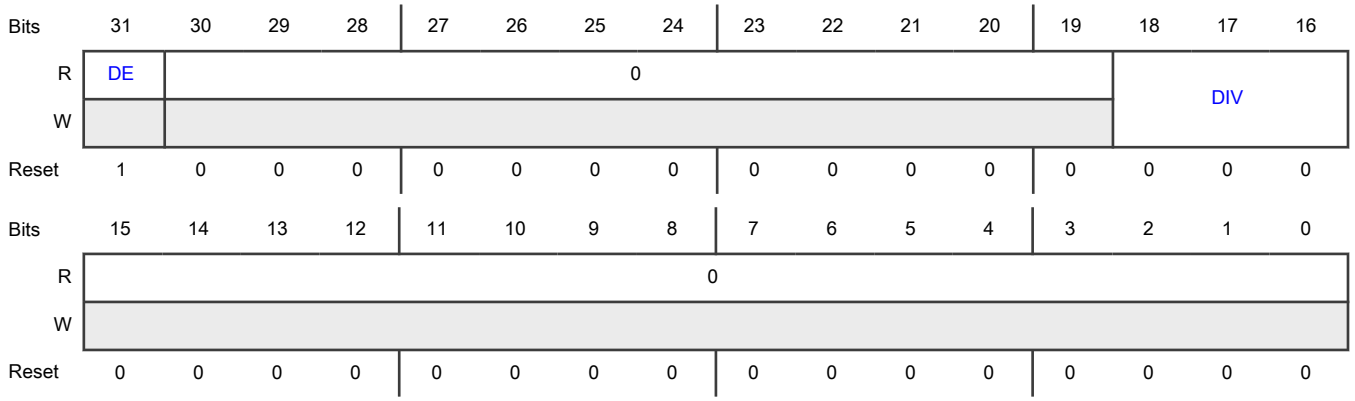
This register controls the clock divider 3 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Reserved 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.12 Clock Mux 0 Divider 4 Control Register (MUX_0_DC_4)

Offset

Register	Offset
MUX_0_DC_4	318h

Function

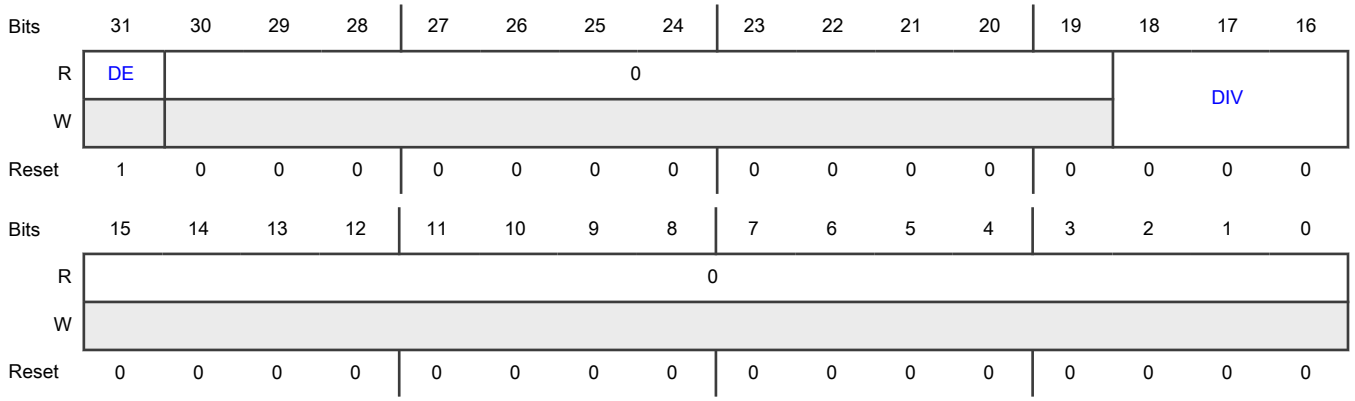
This register controls the clock divider 4 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Reserved 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.13 Clock Mux 0 Divider 5 Control Register (MUX_0_DC_5)

Offset

Register	Offset
MUX_0_DC_5	31Ch

Function

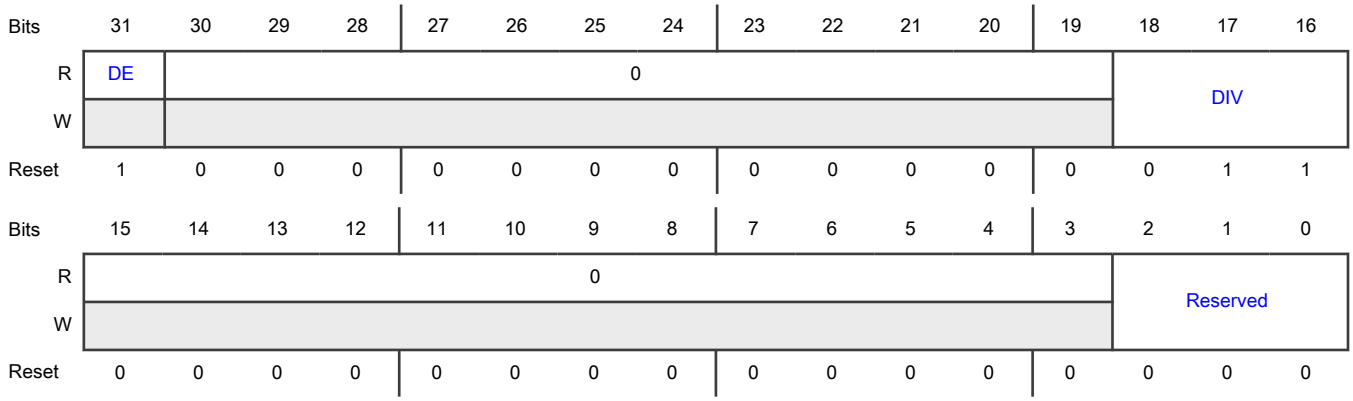
This register controls the clock divider 5 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Reserved 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-3 —	Reserved
2-0 —	Reserved

24.5.14 Clock Mux 0 Divider 6 Control Register (MUX_0_DC_6)

Offset

Register	Offset
MUX_0_DC_6	320h

Function

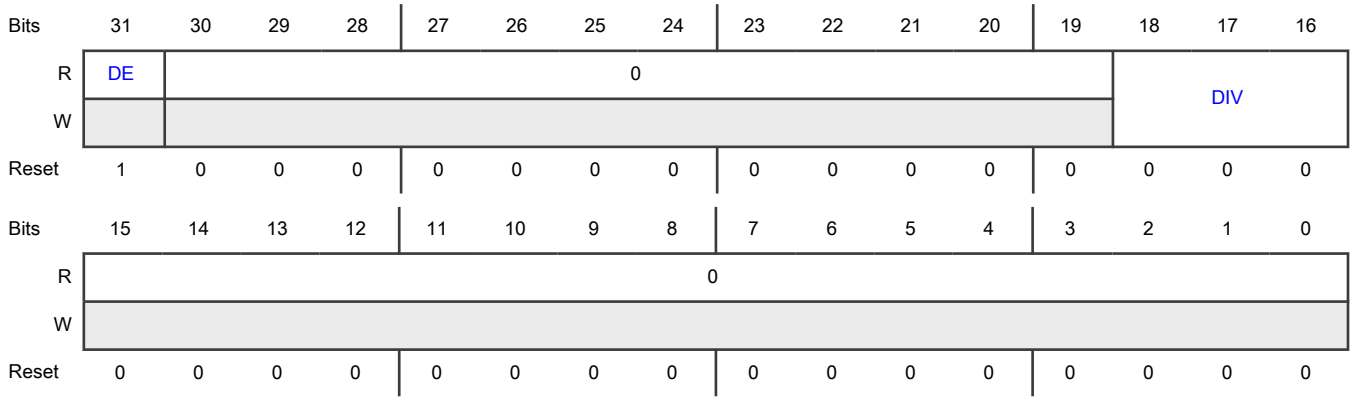
This register controls the clock divider 6 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Reserved 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.15 Clock Mux 0 Divider Trigger Control Register (MUX_0_DIV_TRIG_CTRL)

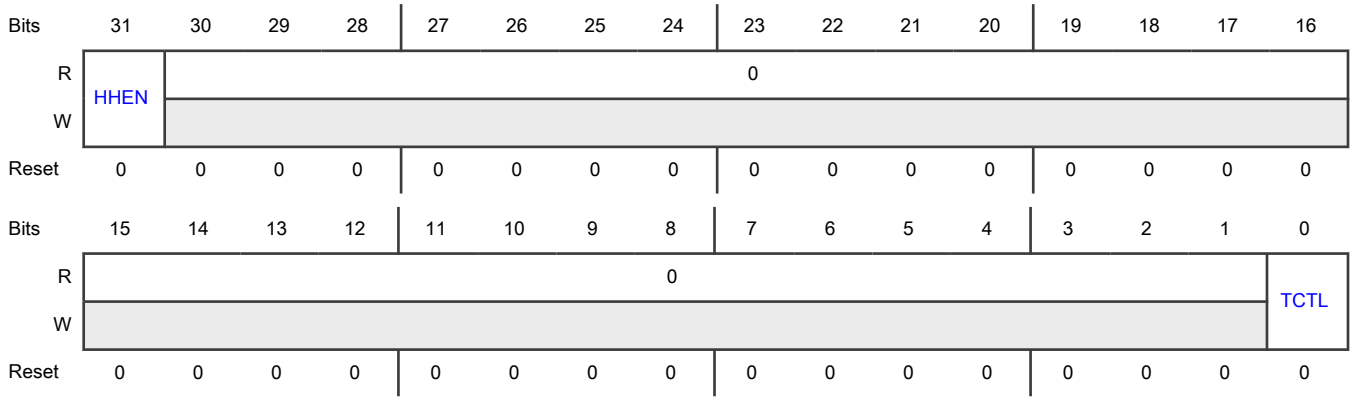
Offset

Register	Offset
MUX_0_DIV_TRIG_CTRL	334h

Function

This register selects whether the dividers associated with clock mux 0 are updated immediately on writing to the corresponding divider configuration register (referred to as immediate divider update) or only on writing to the MC_CGM_MUX_0_DIV_TRIG register (referred to as common trigger update). When common trigger update is configured, this register also controls initiation of the halt handshake protocol with the on-chip AXBS. Software is required to configure HHEN field for handshaking with on-chip AXBS when the ratio of division value among the clock dividers need to be changed.

Diagram



Fields

Field	Function
31 HHEN	Halt handshake enable This field controls the initiation of the halt handshake protocol with AXBS when a common trigger divider update is initiated. 0b - No halt handshake protocol is initiated. 1b - Halt handshake protocol is initiated.
30-1 —	This field is reserved and reads return zeros.
0 TCTL	Trigger control This field controls the divider update configuration between immediate and common update. 0b - Immediate divider update 1b - Common trigger divider update

24.5.16 Clock Mux 0 Divider Trigger Register (MUX_0_DIV_TRIG)

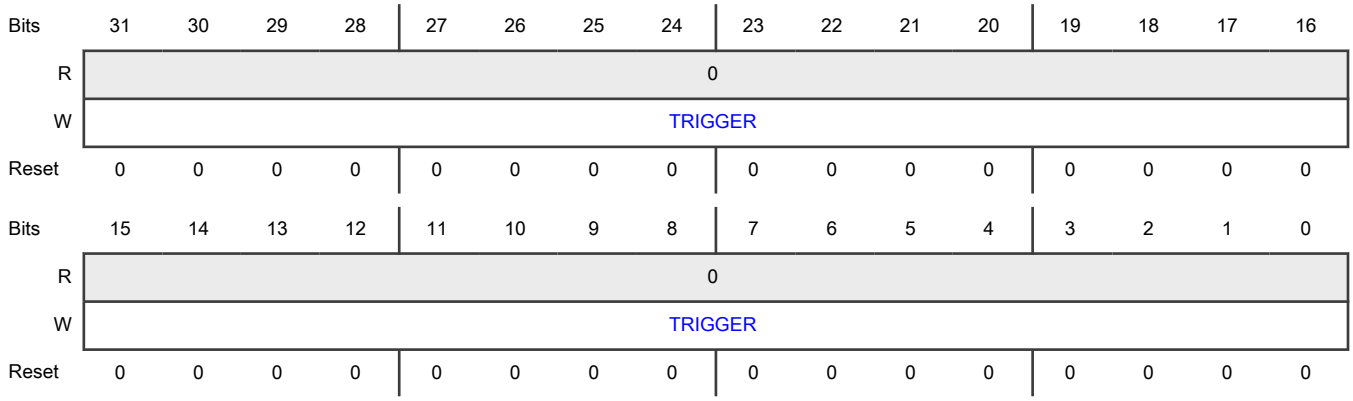
Offset

Register	Offset
MUX_0_DIV_TRIG	338h

Function

This register provides a common trigger for the clock dividers (only 50% duty cycle dividers) of clock mux 0. Writing any value to this register provides a trigger to the dividers. This register should only be written after appropriately configuring the MC_CGM_MUX_0_DIV_TRIG_CTRL register.

Diagram



Fields

Field	Function
31-0 TRIGGER	Trigger for divider update

24.5.17 Clock Mux 0 Divider Update Status Register (MUX_0_DIV_UPD_STAT)

Offset

Register	Offset
MUX_0_DIV_UPD_STAT	33Ch

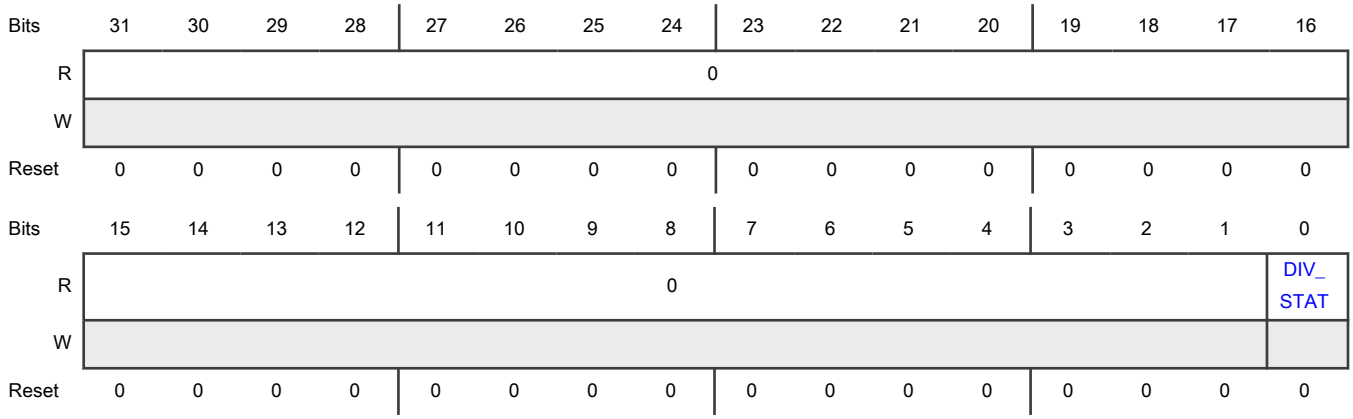
Function

This register provides the update status of the clock dividers corresponding to clock mux 0. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 0 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.18 Clock Mux 1 Select Control Register (MUX_1_CSC)

Offset

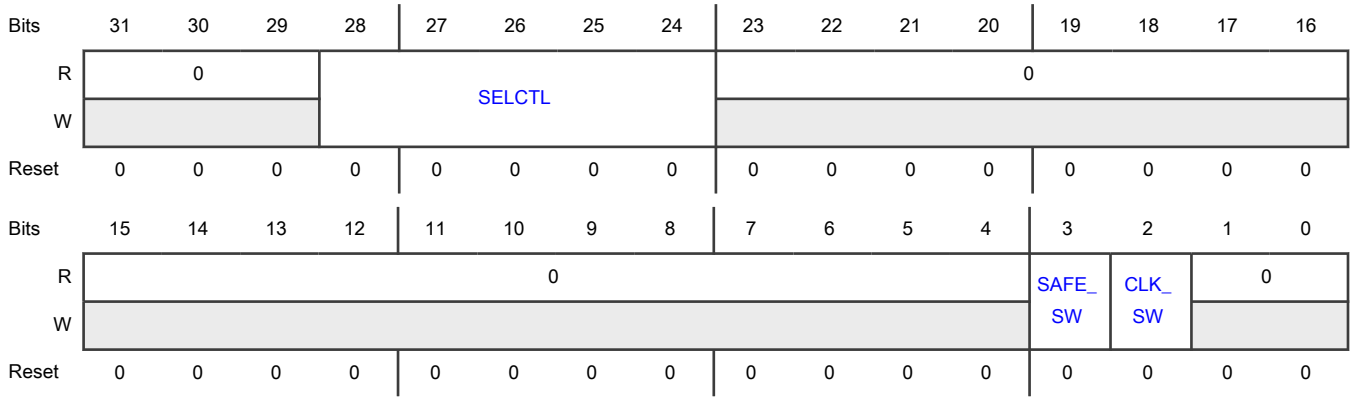
Register	Offset
MUX_1_CSC	340h

Function

This register provides the clock source selection control for clock mux 1. Clock mux 1 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 1. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 1. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.19 Clock Mux 1 Select Status Register (MUX_1_CSS)

Offset

Register	Offset
MUX_1_CSS	344h

Function

This register provides the current clock source selection status for clock mux 1.

This register is reset on destructive reset only.

Diagram

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

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

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 1. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 1. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 1. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.20 Clock Mux 1 Divider 0 Control Register (MUX_1_DC_0)

Offset

Register	Offset
MUX_1_DC_0	348h

Function

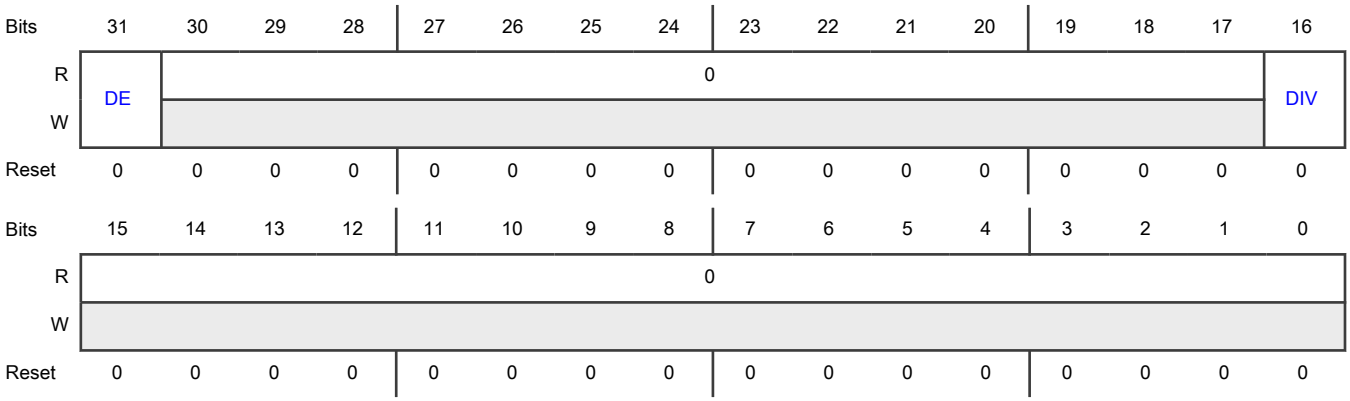
This register controls the clock divider 0 for clock mux 1.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-17 —	This field is reserved and reads return zeros.
16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.21 Clock Mux 1 Divider Update Status Register (MUX_1_DIV_UPD_STAT)

Offset

Register	Offset
MUX_1_DIV_UPD_STAT	37Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 1. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

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

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

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 1 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.22 Clock Mux 2 Select Control Register (MUX_2_CSC)

Offset

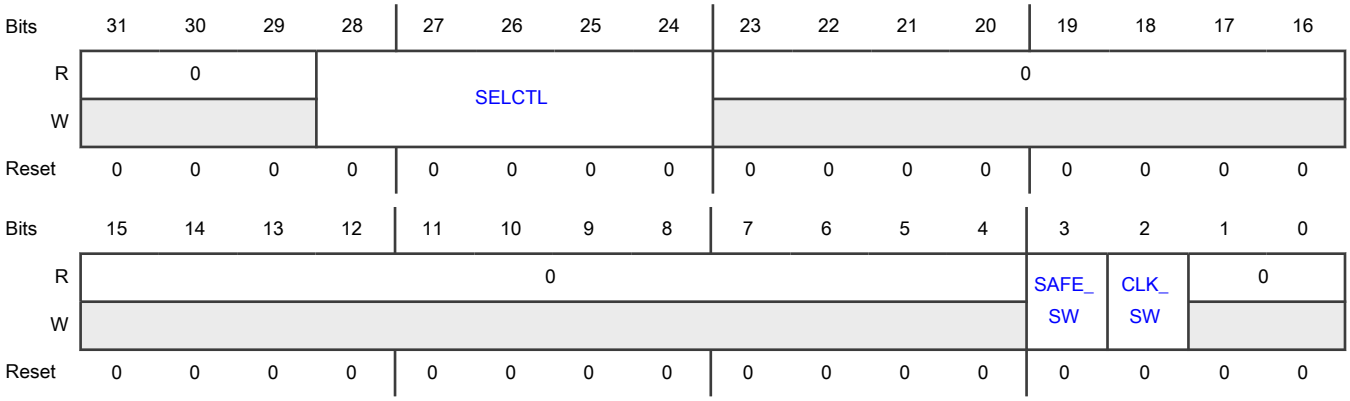
Register	Offset
MUX_2_CSC	380h

Function

This register provides the clock source selection control for clock mux 2. Clock mux 2 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 2. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 2. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.23 Clock Mux 2 Select Status Register (MUX_2_CSS)

Offset

Register	Offset
MUX_2_CSS	384h

Function

This register provides the current clock source selection status for clock mux 2.

This register is reset on destructive reset only.

Diagram

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

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

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 2. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 2. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 2. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.24 Clock Mux 2 Divider 0 Control Register (MUX_2_DC_0)

Offset

Register	Offset
MUX_2_DC_0	388h

Function

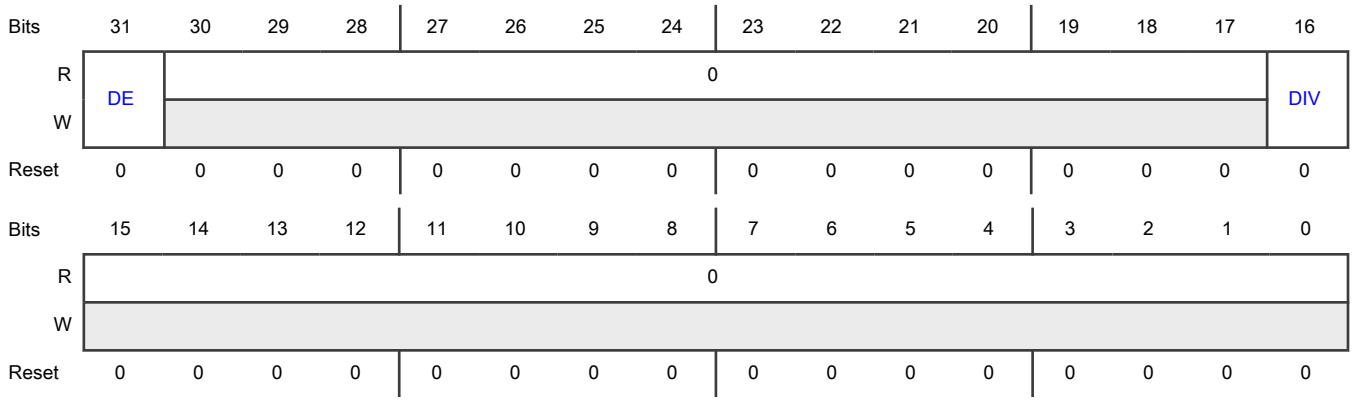
This register controls the clock divider 0 for clock mux 2.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-17 —	This field is reserved and reads return zeros.
16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.25 Clock Mux 2 Divider Update Status Register (MUX_2_DIV_UPD_STAT)

Offset

Register	Offset
MUX_2_DIV_UPD_STAT	3BCh

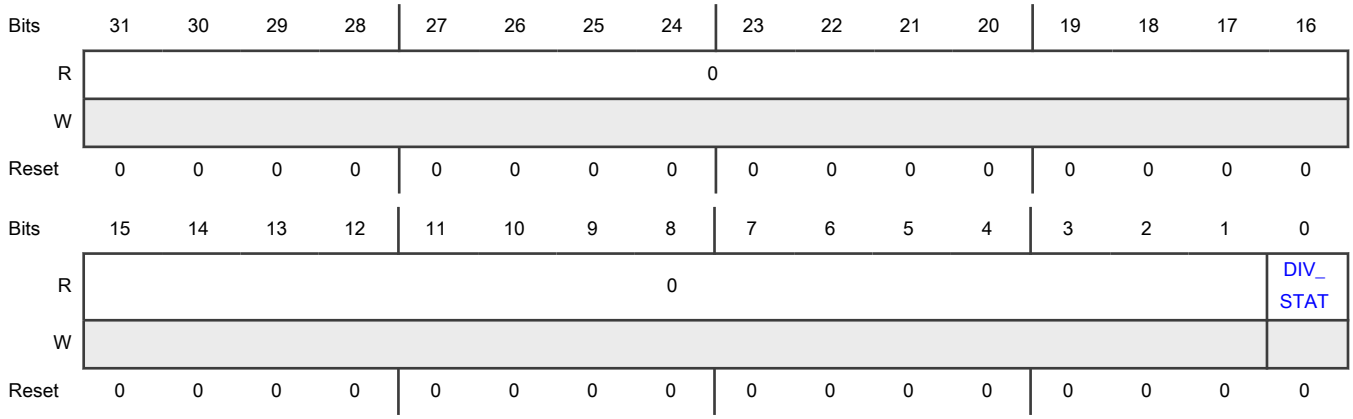
Function

This register provides the update status of the clock dividers corresponding to clock mux 2. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 2 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.26 Clock Mux 3 Select Control Register (MUX_3_CSC)

Offset

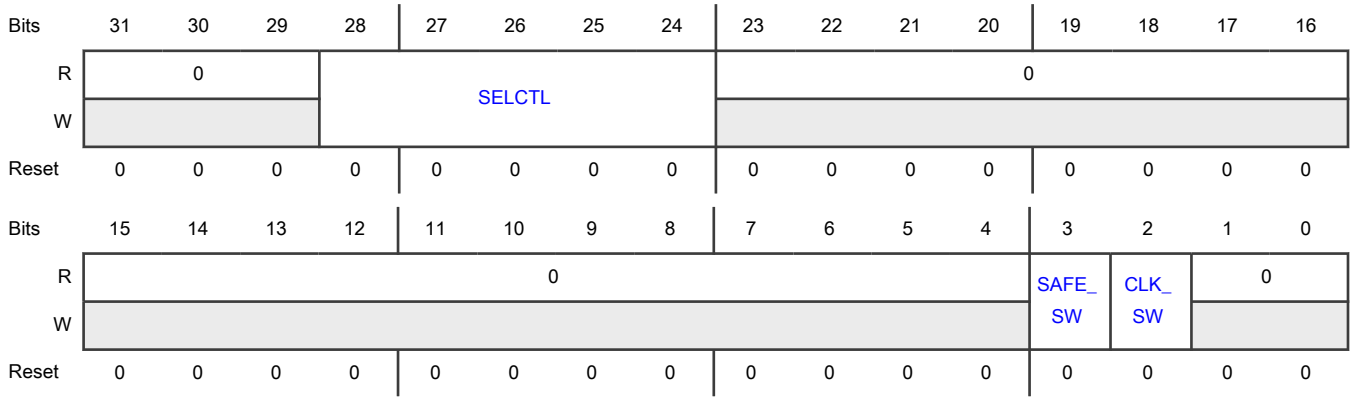
Register	Offset
MUX_3_CSC	3C0h

Function

This register provides the clock source selection control for clock mux 3. Clock mux 3 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 3. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 3. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.27 Clock Mux 3 Select Status Register (MUX_3_CSS)

Offset

Register	Offset
MUX_3_CSS	3C4h

Function

This register provides the current clock source selection status for clock mux 3.

This register is reset on destructive reset only.

Diagram

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

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

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 3. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 3. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 3. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.28 Clock Mux 3 Divider 0 Control Register (MUX_3_DC_0)

Offset

Register	Offset
MUX_3_DC_0	3C8h

Function

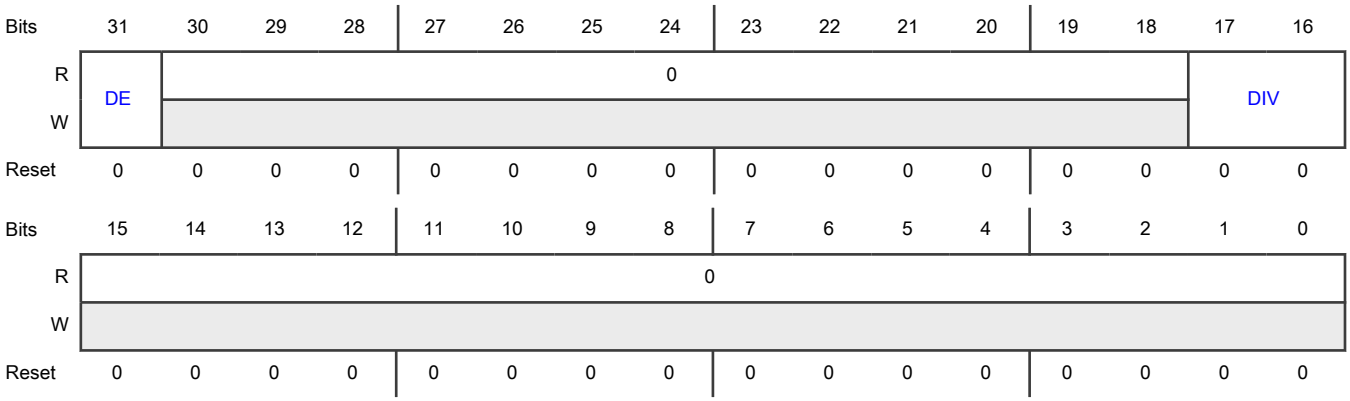
This register controls the clock divider 0 for clock mux 3.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-18 —	This field is reserved and reads return zeros.
17-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.29 Clock Mux 3 Divider Update Status Register (MUX_3_DIV_UPD_STAT)

Offset

Register	Offset
MUX_3_DIV_UPD_STAT	3FCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 3. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

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

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

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 3 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.30 Clock Mux 4 Select Control Register (MUX_4_CSC)

Offset

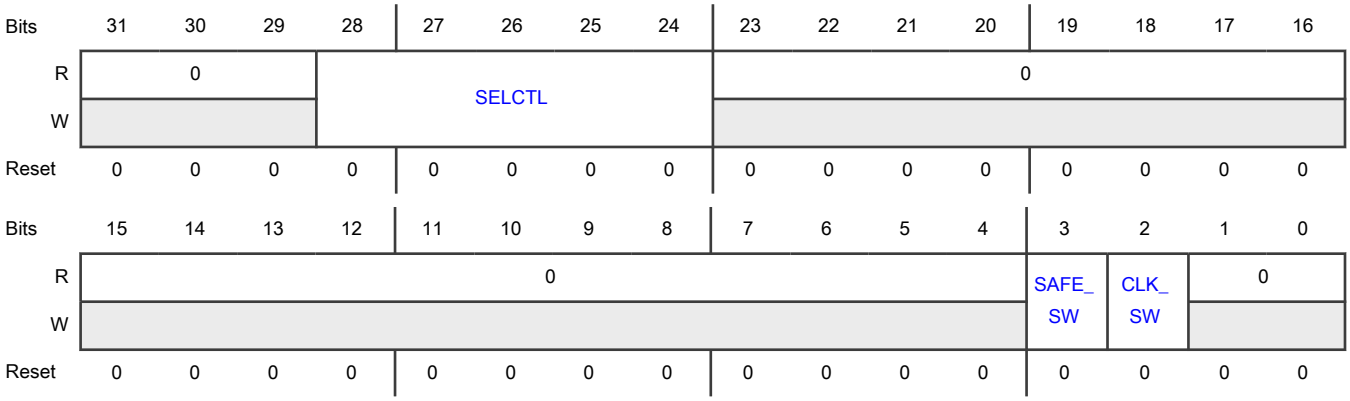
Register	Offset
MUX_4_CSC	400h

Function

This register provides the clock source selection control for clock mux 4. Clock mux 4 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 4. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 4. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.31 Clock Mux 4 Select Status Register (MUX_4_CSS)

Offset

Register	Offset
MUX_4_CSS	404h

Function

This register provides the current clock source selection status for clock mux 4.

This register is reset on destructive reset only.

Diagram

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

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

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 4. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 4. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 4. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.32 Clock Mux 4 Divider 0 Control Register (MUX_4_DC_0)

Offset

Register	Offset
MUX_4_DC_0	408h

Function

This register controls the clock divider 0 for clock mux 4.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

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

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

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-18 —	This field is reserved and reads return zeros.
17-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.33 Clock Mux 4 Divider Update Status Register (MUX_4_DIV_UPD_STAT)

Offset

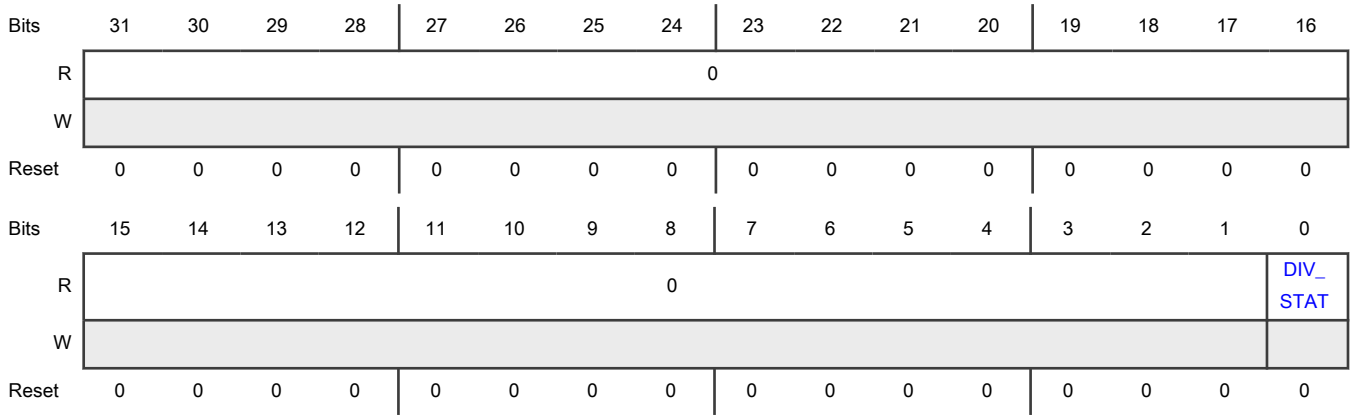
Register	Offset
MUX_4_DIV_UPD_STAT	43Ch

Function
This register provides the update status of the clock dividers corresponding to clock mux 4. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 4 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.34 Clock Mux 5 Select Control Register (MUX_5_CSC)

Offset

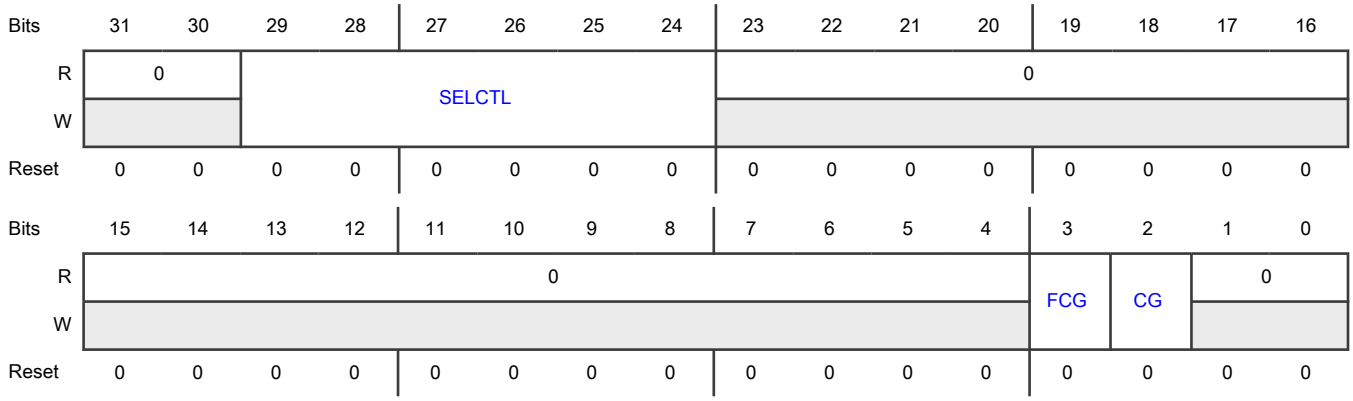
Register	Offset
MUX_5_CSC	440h

Function

This register provides the clock source selection control of clock mux 5. Clock mux 5 implements software control clock switching, and a graceful clock switch can be performed by executing a sequence of steps in software. See "Software-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 5. The reserved values are not displayed. 00_0000b - FIRC 00_0001b - SIRC 00_0010b - FXOSC 00_0100b - SXOSC 01_0111b - AIPS_SLOW_CLK 11_0010b - Reserved
23-4 —	This field is reserved and reads return zeros.
3 FCG	Force clock gate Writing 1 to this bit gates the clock at the output of clock mux 5 to logic-0 irrespective of the logic level of the currently selected clock. Clock gating using this bit should only be performed when it is insured that current clock source is inactive.
2 CG	Clock gate Writing 1 to this bit gates the clock at the output of clock mux 5 to logic-0. Using this bit it is insured that no glitches are resulted when gating the clock.
1-0 —	This field is reserved and reads return zeros.

24.5.35 Clock Mux 5 Select Status Register (MUX_5_CSS)

Offset

Register	Offset
MUX_5_CSS	444h

Function

This register provides the current clock source selection status for clock mux 5.

This register is reset on destructive reset only.

Diagram

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

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

Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 5. The reserved values are not displayed. 00_0000b - FIRC 00_0001b - SIRC 00_0010b - FXOSC 00_0100b - SXOSC 01_0111b - AIPS_SLOW_CLK 11_0010b - Reserved
23-20 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-18 —	This field is reserved and reads return zeros.
17 CS	Clock status This field indicates state of the clock at the output of the clock mux. 0b - Clock is gated to logic-0 at output of clock mux 1b - Clock mux is transparent. Active clock pulses at input of clock mux results in same number of pulses at its output
16 GRIP	Gating request is in progress. When a clock gate request is given this bit indicates if the clock gating at the output of mux has completed or not. 0b - Clock source gating or ungating has completed. 1b - Clock source gating or ungating is in progress.
15-0 —	This field is reserved and reads return zeros.

24.5.36 Clock Mux 5 Divider 0 Control Register (MUX_5_DC_0)

Offset

Register	Offset
MUX_5_DC_0	448h

Function

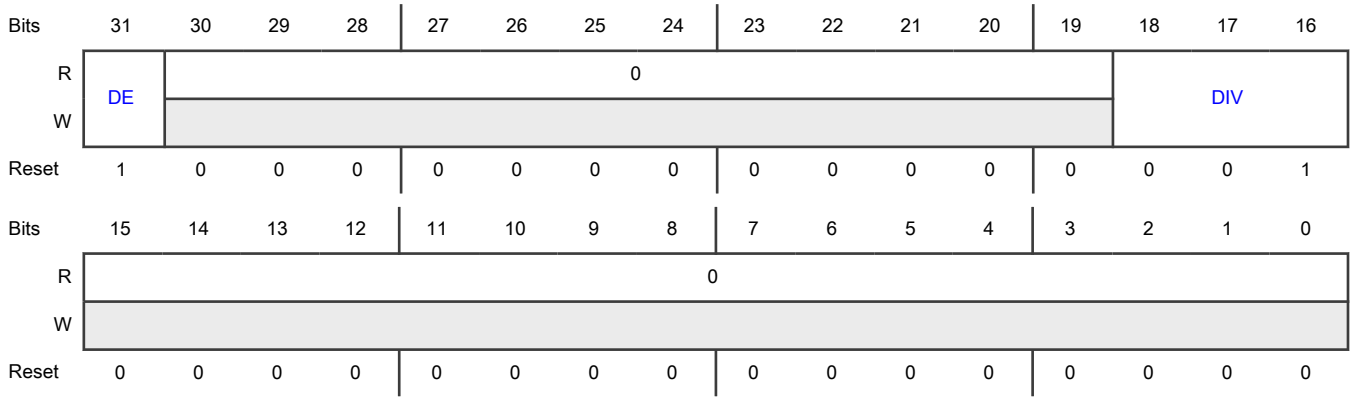
This register controls the clock divider 0 for clock mux 5.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.37 Clock Mux 5 Divider Update Status Register (MUX_5_DIV_UPD_STAT)

Offset

Register	Offset
MUX_5_DIV_UPD_STAT	47Ch

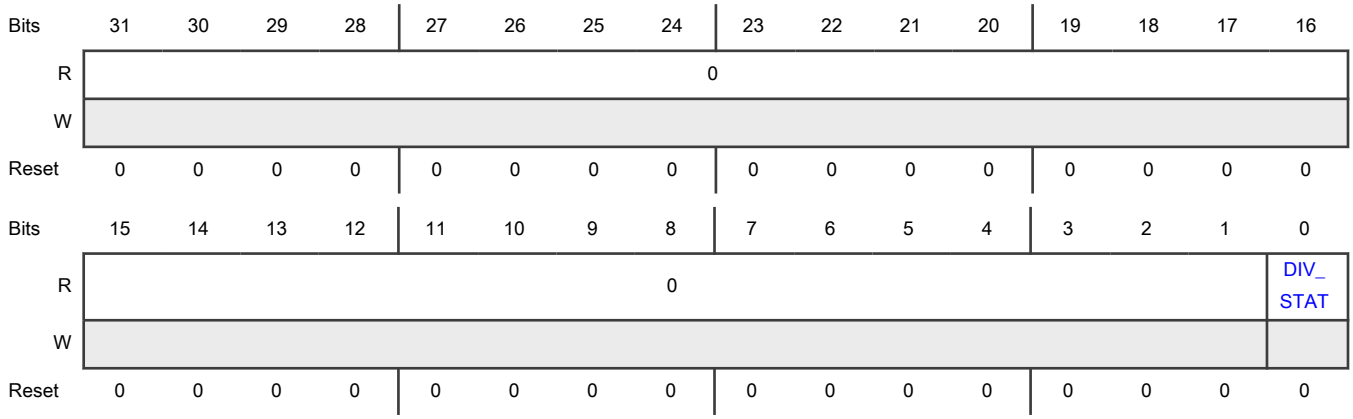
Function

This register provides the update status of the clock dividers corresponding to clock mux 5. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 5 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.38 Clock Mux 6 Select Control Register (MUX_6_CSC)

Offset

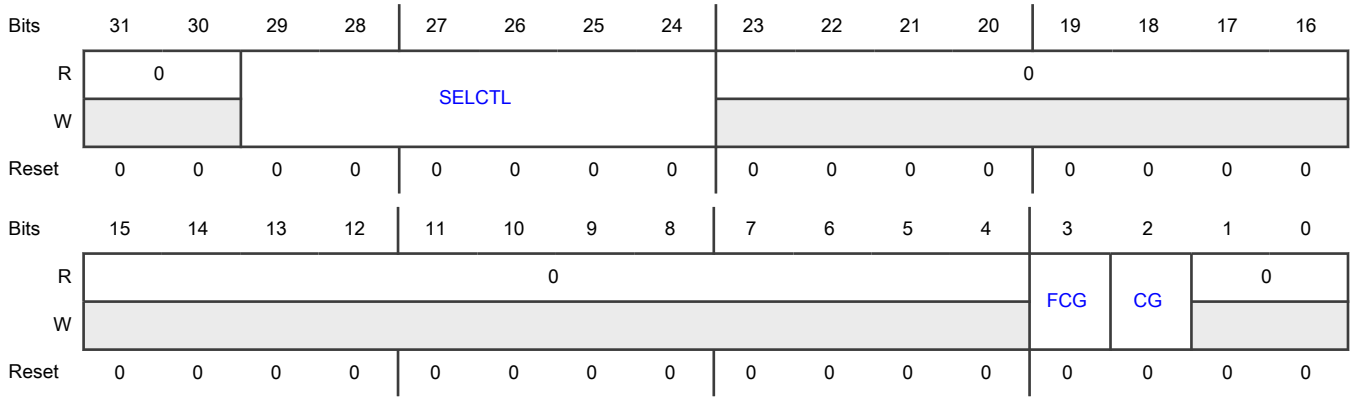
Register	Offset
MUX_6_CSC	480h

Function

This register provides the clock source selection control of clock mux 6. Clock mux 6 implements software control clock switching, and a graceful clock switch can be performed by executing a sequence of steps in software. See "Software-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 6. The reserved values are not displayed. 00_0000b - FIRC 00_0001b - SIRC 00_0010b - FXOSC 00_0100b - SXOSC 00_1000b - PLL_PHI0_CLK 00_1001b - PLL_PHI1_CLK 01_0000b - CORE_CLK 01_0011b - HSE_CLK 01_0110b - AIPS_PLAT_CLK 01_0111b - AIPS_SLOW_CLK 01_1000b - EMAC_RMII_TX_CLK 01_1001b - EMAC_RX_CLK 11_0011b - Reserved 11_0100b - Reserved 11_0101b - Reserved 11_0110b - Reserved 11_0111b - Reserved 11_1000b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11_1001b - Reserved 11_1010b - Reserved 11_1011b - Reserved 11_1100b - Reserved 11_1101b - Reserved 11_1110b - Reserved
23-4 —	This field is reserved and reads return zeros.
3 FCG	Force clock gate Writing 1 to this bit gates the clock at the output of clock mux 6 to logic-0 irrespective of the logic level of the currently selected clock. Clock gating using this bit should only be performed when it is insured that current clock source is inactive.
2 CG	Clock gate Writing 1 to this bit gates the clock at the output of clock mux 6 to logic-0. Using this bit it is insured that no glitches are resulted when gating the clock.
1-0 —	This field is reserved and reads return zeros.

24.5.39 Clock Mux 6 Select Status Register (MUX_6_CSS)

Offset

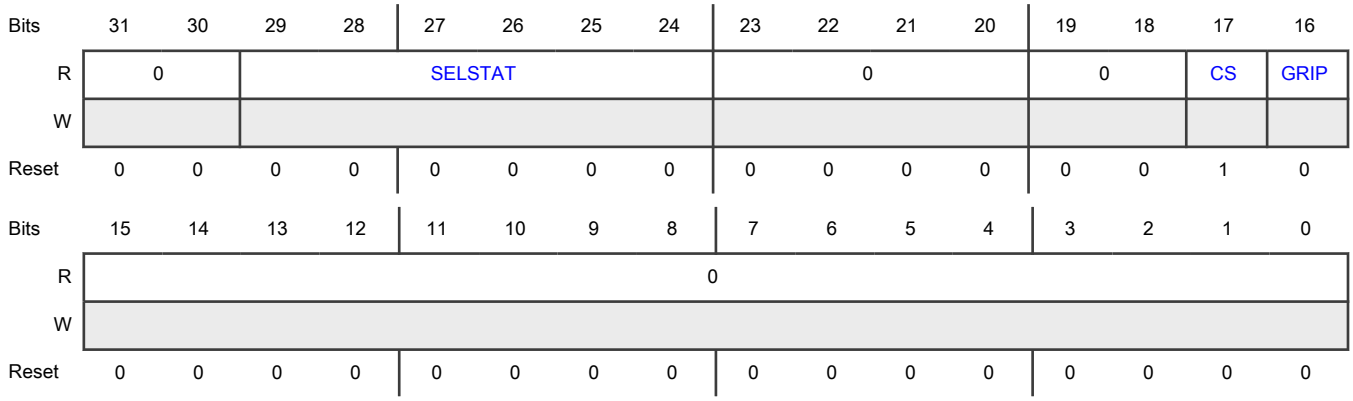
Register	Offset
MUX_6_CSS	484h

Function

This register provides the current clock source selection status for clock mux 6.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 6. The reserved values are not displayed. 00_0000b - FIRC 00_0001b - SIRC 00_0010b - FXOSC 00_0100b - SXOSC 00_1000b - PLL_PHI0_CLK 00_1001b - PLL_PHI1_CLK 01_0000b - CORE_CLK 01_0011b - HSE_CLK 01_0110b - AIPS_PLAT_CLK 01_0111b - AIPS_SLOW_CLK 01_1000b - EMAC_RMII_TX_CLK 01_1001b - EMAC_RX_CLK 11_0011b - Reserved 11_0100b - Reserved 11_0101b - Reserved 11_0110b - Reserved 11_0111b - Reserved 11_1000b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11_1001b - Reserved 11_1010b - Reserved 11_1011b - Reserved 11_1100b - Reserved 11_1101b - Reserved 11_1110b - Reserved
23-20 —	This field is reserved and reads return zeros.
19-18 —	This field is reserved and reads return zeros.
17 CS	Clock status This field indicates state of the clock at the output of the clock mux. 0b - Clock is gated to logic-0 at output of clock mux 1b - Clock mux is transparent. Active clock pulses at input of clock mux results in same number of pulses at its output
16 GRIP	Gating request is in progress. When a clock gate request is given this bit indicates if the clock gating at the output of mux has completed or not. 0b - Clock source gating or ungating has completed. 1b - Clock source gating or ungating is in progress.
15-0 —	This field is reserved and reads return zeros.

24.5.40 Clock Mux 6 Divider 0 Control Register (MUX_6_DC_0)

Offset

Register	Offset
MUX_6_DC_0	488h

Function

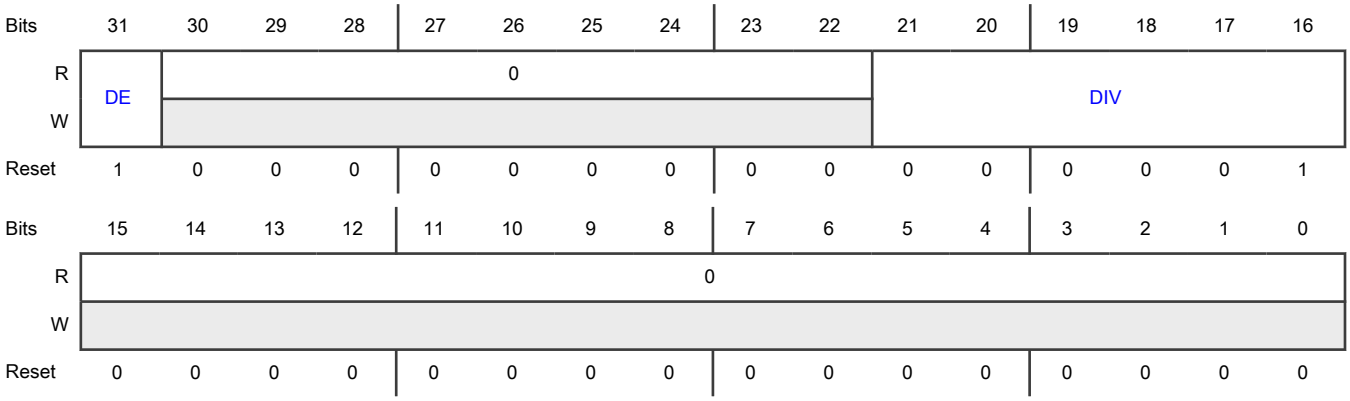
This register controls the clock divider 0 for clock mux 6.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.41 Clock Mux 6 Divider Update Status Register (MUX_6_DIV_UPD_STAT)

Offset

Register	Offset
MUX_6_DIV_UPD_STAT	4BCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 6. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other

clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

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

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

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 6 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.42 Clock Mux 7 Select Control Register (MUX_7_CSC)

Offset

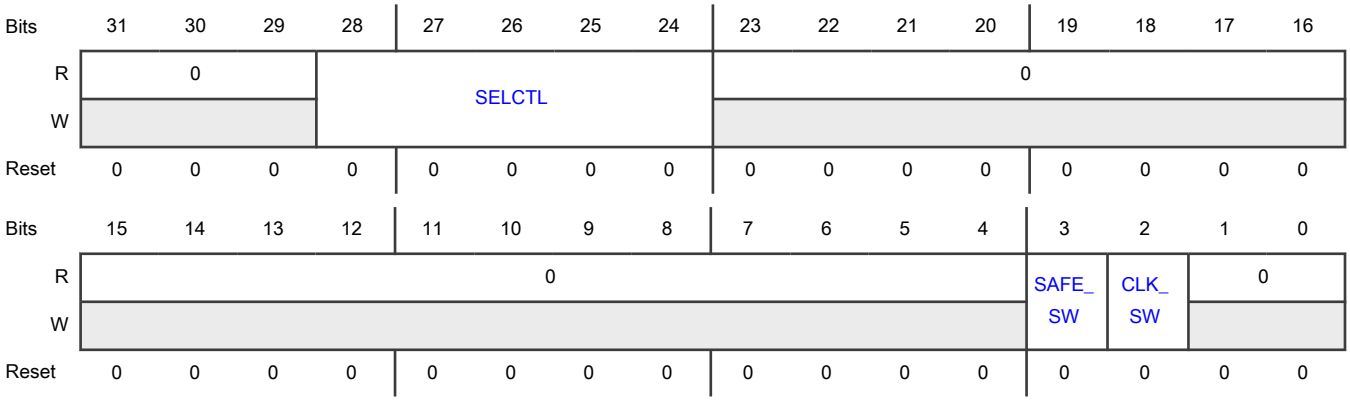
Register	Offset
MUX_7_CSC	4C0h

Function

This register provides the clock source selection control for clock mux 7. Clock mux 7 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 7. The reserved values are not displayed. 0_0000b - FIRC 1_1000b - EMAC_RMII_TX_CLK 1_1001b - EMAC_RX_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 7. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.43 Clock Mux 7 Select Status Register (MUX_7_CSS)

Offset

Register	Offset
MUX_7_CSS	4C4h

Function

This register provides the current clock source selection status for clock mux 7.

This register is reset on destructive reset only.

Diagram

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

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

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 7. The reserved values are not displayed. 0_0000b - FIRC 1_1000b - EMAC_RMII_TX_CLK 1_1001b - EMAC_RX_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 7. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 7. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.44 Clock Mux 7 Divider 0 Control Register (MUX_7_DC_0)

Offset

Register	Offset
MUX_7_DC_0	4C8h

Function

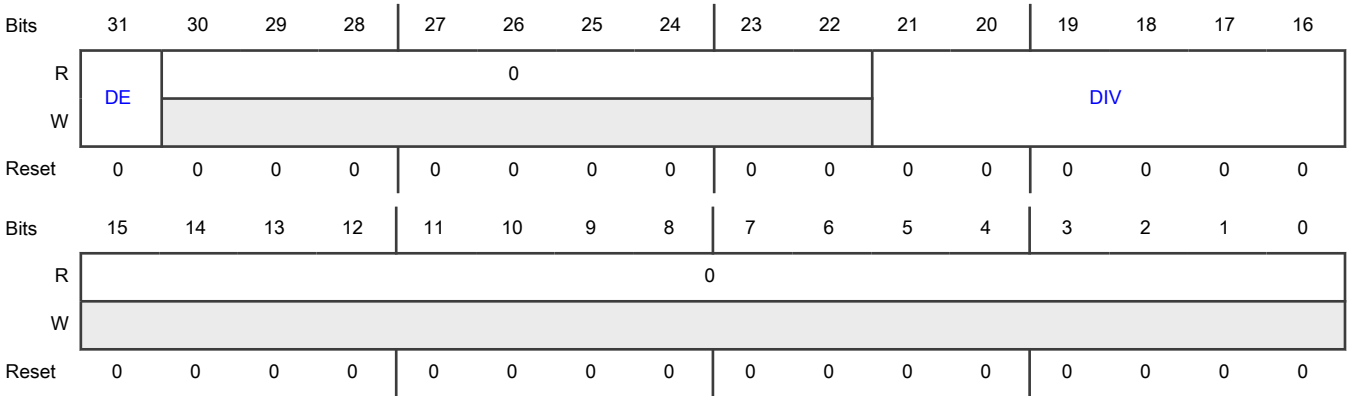
This register controls the clock divider 0 for clock mux 7.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.45 Clock Mux 7 Divider Update Status Register (MUX_7_DIV_UPD_STAT)

Offset

Register	Offset
MUX_7_DIV_UPD_STAT	4FCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 7. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

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

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

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 7 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.46 Clock Mux 8 Select Control Register (MUX_8_CSC)

Offset

Register	Offset
MUX_8_CSC	500h

Function

This register provides the clock source selection control for clock mux 8. Clock mux 8 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram

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

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

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 8. The reserved values are not displayed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0_0000b - FIRC 1_1000b - EMAC_RMII_TX_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 8. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.47 Clock Mux 8 Select Status Register (MUX_8_CSS)

Offset

Register	Offset
MUX_8_CSS	504h

Function

This register provides the current clock source selection status for clock mux 8.

This register is reset on destructive reset only.

Diagram

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

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

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 8. The reserved values are not displayed. 0_0000b - FIRC 1_1000b - EMAC_RMII_TX_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
SAFE_SW	This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 8. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 8. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.48 Clock Mux 8 Divider 0 Control Register (MUX_8_DC_0)

Offset

Register	Offset
MUX_8_DC_0	508h

Function

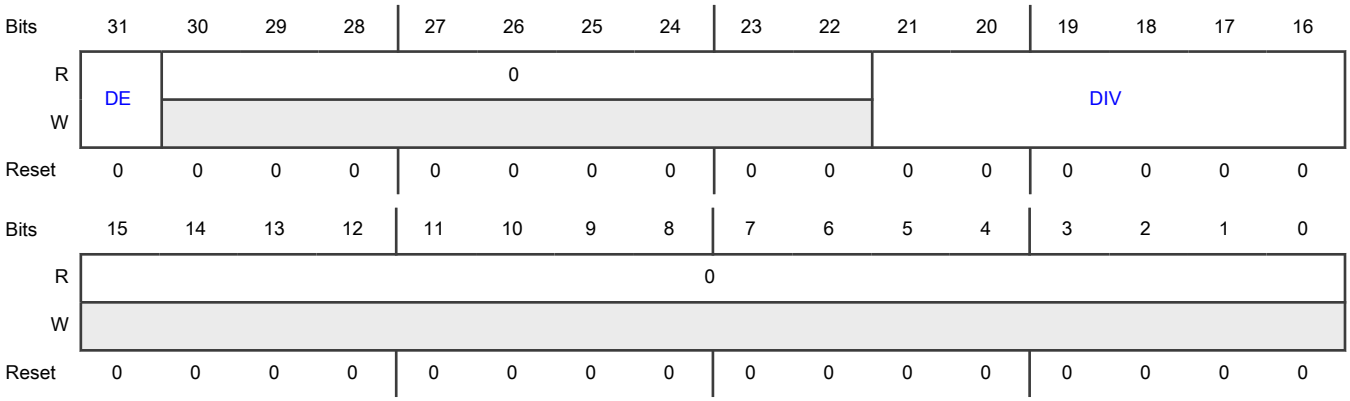
This register controls the clock divider 0 for clock mux 8.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.49 Clock Mux 8 Divider Update Status Register (MUX_8_DIV_UPD_STAT)

Offset

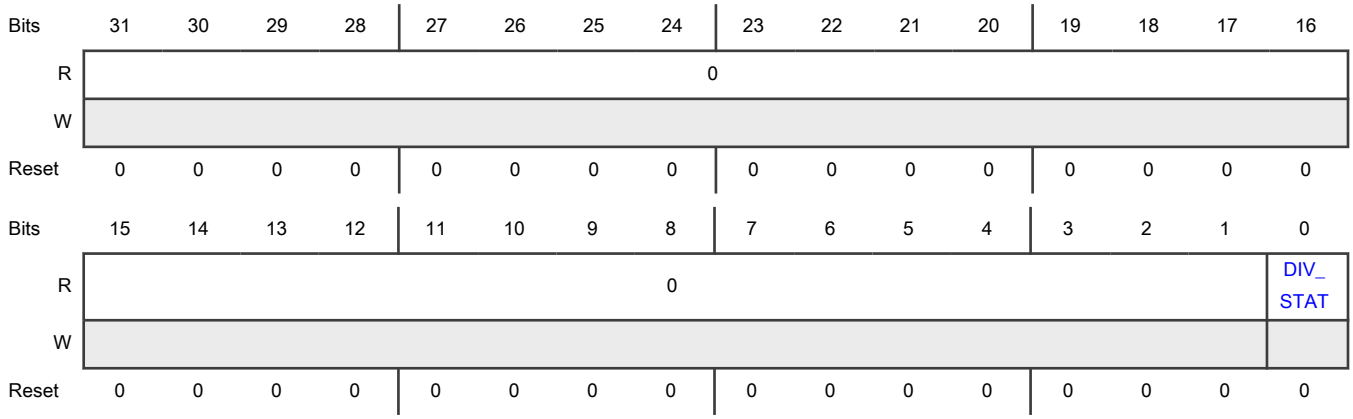
Register	Offset
MUX_8_DIV_UPD_STAT	53Ch

Function
This register provides the update status of the clock dividers corresponding to clock mux 8. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 8 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.50 Clock Mux 9 Select Control Register (MUX_9_CSC)

Offset

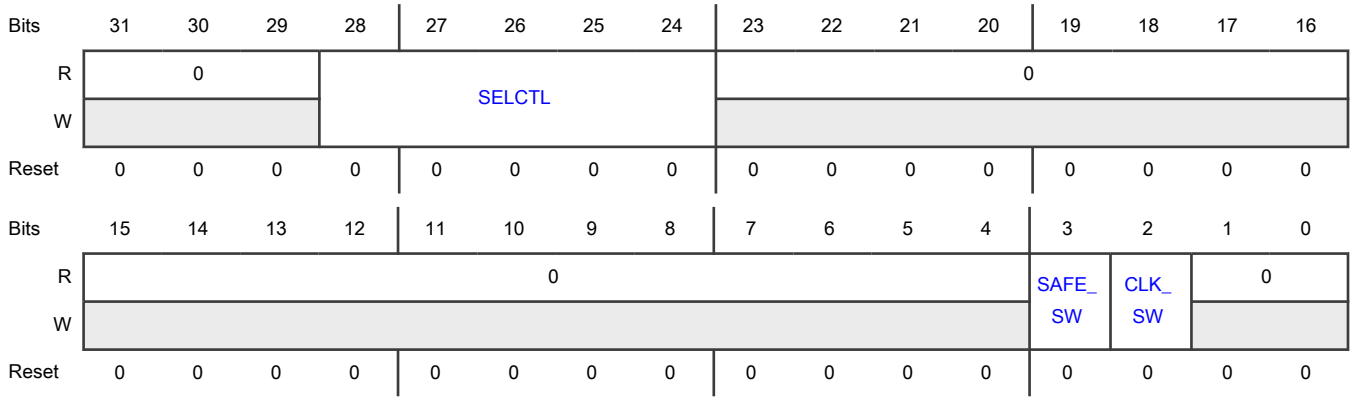
Register	Offset
MUX_9_CSC	540h

Function

This register provides the clock source selection control for clock mux 9. Clock mux 9 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 9. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 0_1000b - PLL_PHI0_CLK 1_1000b - EMAC_RMII_TX_CLK 1_1001b - EMAC_RX_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 9. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.51 Clock Mux 9 Select Status Register (MUX_9_CSS)

Offset

Register	Offset
MUX_9_CSS	544h

Function

This register provides the current clock source selection status for clock mux 9.

This register is reset on destructive reset only.

Diagram

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

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

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 9. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 0_1000b - PLL_PHI0_CLK 1_1000b - EMAC_RMII_TX_CLK 1_1001b - EMAC_RX_CLK
23-20 —	This field is reserved and reads return zeros.
19-17	Switch trigger cause

Table continues on the next page...

Table continued from the previous page...

Field	Function
SWTRG	This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 9. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 9. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

24.5.52 Clock Mux 9 Divider 0 Control Register (MUX_9_DC_0)

Offset

Register	Offset
MUX_9_DC_0	548h

Function

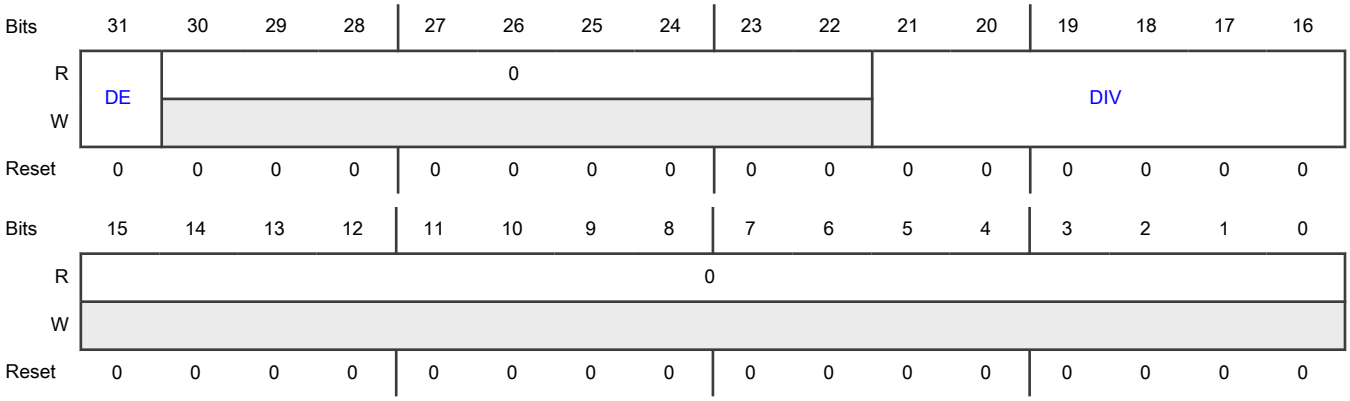
This register controls the clock divider 0 for clock mux 9.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.53 Clock Mux 9 Divider Update Status Register (MUX_9_DIV_UPD_STAT)

Offset

Register	Offset
MUX_9_DIV_UPD_STAT	57Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 9. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

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

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

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 9 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.54 Clock Mux 10 Select Control Register (MUX_10_CSC)

Offset

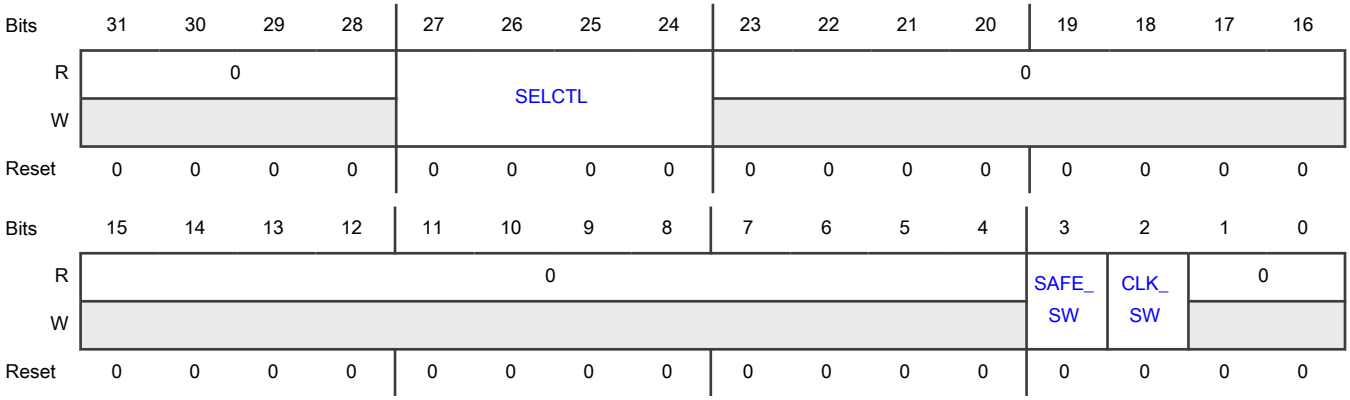
Register	Offset
MUX_10_CSC	580h

Function

This register provides the clock source selection control for clock mux 10. Clock mux 10 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELCTL	Clock source selection control Selects the source clock for clock mux 10. The reserved values are not displayed. 0000b - FIRC 0010b - FXOSC 1001b - PLL_PHI1_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 10. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.55 Clock Mux 10 Select Status Register (MUX_10_CSS)

Offset

Register	Offset
MUX_10_CSS	584h

Function

This register provides the current clock source selection status for clock mux 10.

This register is reset on destructive reset only.

Diagram

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

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

Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 10. The reserved values are not displayed. 0000b - FIRC 0010b - FXOSC 1001b - PLL_PHI1_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 10. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 10. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.56 Clock Mux 10 Divider 0 Control Register (MUX_10_DC_0)

Offset

Register	Offset
MUX_10_DC_0	588h

Function

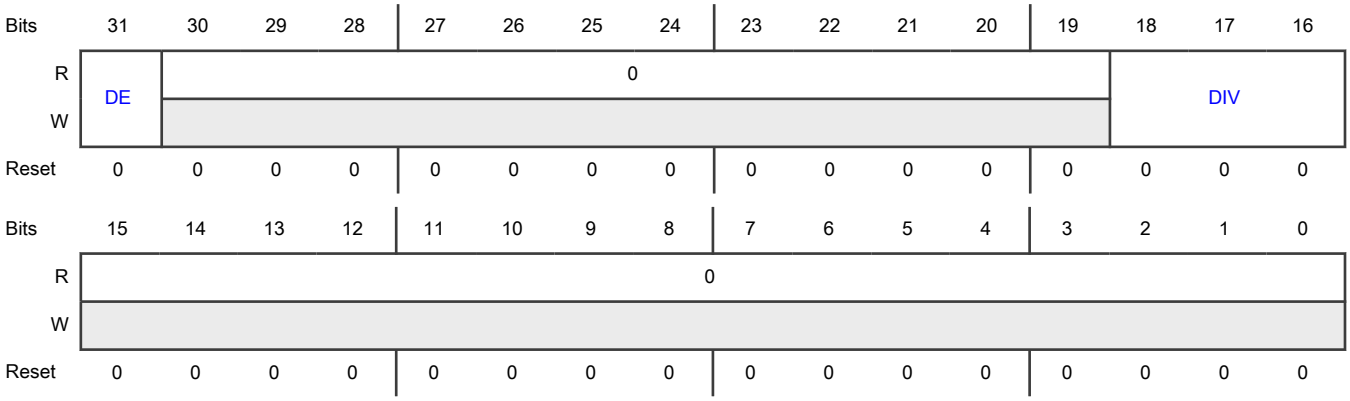
This register controls the clock divider 0 for clock mux 10.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.57 Clock Mux 10 Divider Update Status Register (MUX_10_DIV_UPD_STAT)

Offset

Register	Offset
MUX_10_DIV_UPD_STAT	5BCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 10. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other

clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

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

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

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 10 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.58 Clock Mux 11 Select Control Register (MUX_11_CSC)

Offset

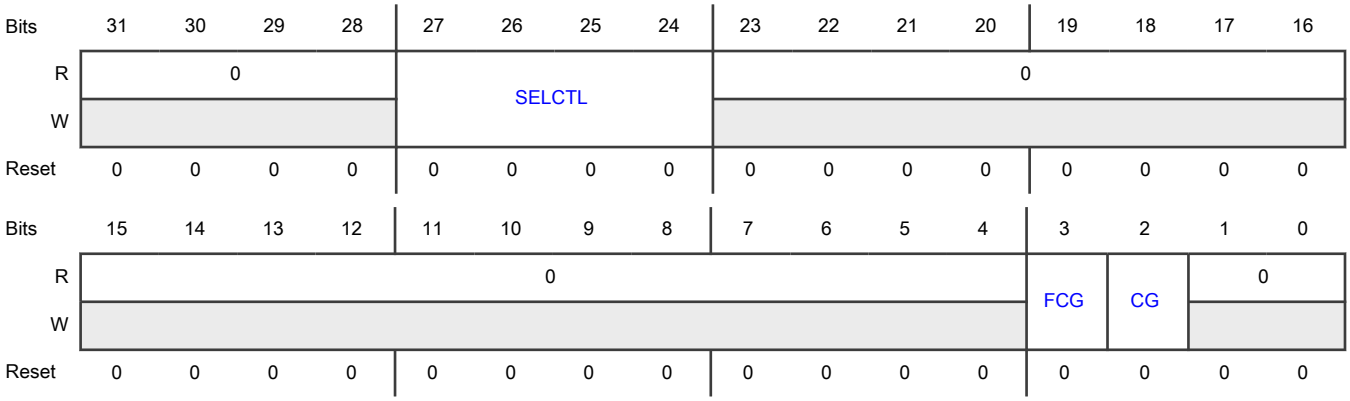
Register	Offset
MUX_11_CSC	5C0h

Function

This register provides the clock source selection control of clock mux 11. Clock mux 11 implements software control clock switching, and a graceful clock switch can be performed by executing a sequence of steps in software. See "Software-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELCTL	Clock source selection control Selects the source clock for clock mux 11. The reserved values are not displayed. 0000b - FIRC 0010b - FXOSC 1000b - PLL_PHI0_CLK 1001b - PLL_PHI1_CLK
23-4 —	This field is reserved and reads return zeros.
3 FCG	Force clock gate Writing 1 to this bit gates the clock at the output of clock mux 11 to logic-0 irrespective of the logic level of the currently selected clock. Clock gating using this bit should only be performed when it is insured that current clock source is inactive.
2 CG	Clock gate Writing 1 to this bit gates the clock at the output of clock mux 11 to logic-0. Using this bit it is insured that no glitches are resulted when gating the clock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1-0 —	This field is reserved and reads return zeros.

24.5.59 Clock Mux 11 Select Status Register (MUX_11_CSS)

Offset

Register	Offset
MUX_11_CSS	5C4h

Function

This register provides the current clock source selection status for clock mux 11.

This register is reset on destructive reset only.

Diagram

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

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

Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 11. The reserved values are not displayed. 0000b - FIRC 0010b - FXOSC

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1000b - PLL_PHI0_CLK 1001b - PLL_PHI1_CLK
23-20 —	This field is reserved and reads return zeros.
19-18 —	This field is reserved and reads return zeros.
17 CS	Clock status This field indicates state of the clock at the output of the clock mux. 0b - Clock is gated to logic-0 at output of clock mux 1b - Clock mux is transparent. Active clock pulses at input of clock mux results in same number of pulses at its output
16 GRIP	Gating request is in progress. When a clock gate request is given this bit indicates if the clock gating at the output of mux has completed or not. 0b - Clock source gating or ungating has completed. 1b - Clock source gating or ungating is in progress.
15-0 —	This field is reserved and reads return zeros.

24.5.60 Clock Mux 11 Divider 0 Control Register (MUX_11_DC_0)

Offset

Register	Offset
MUX_11_DC_0	5C8h

Function

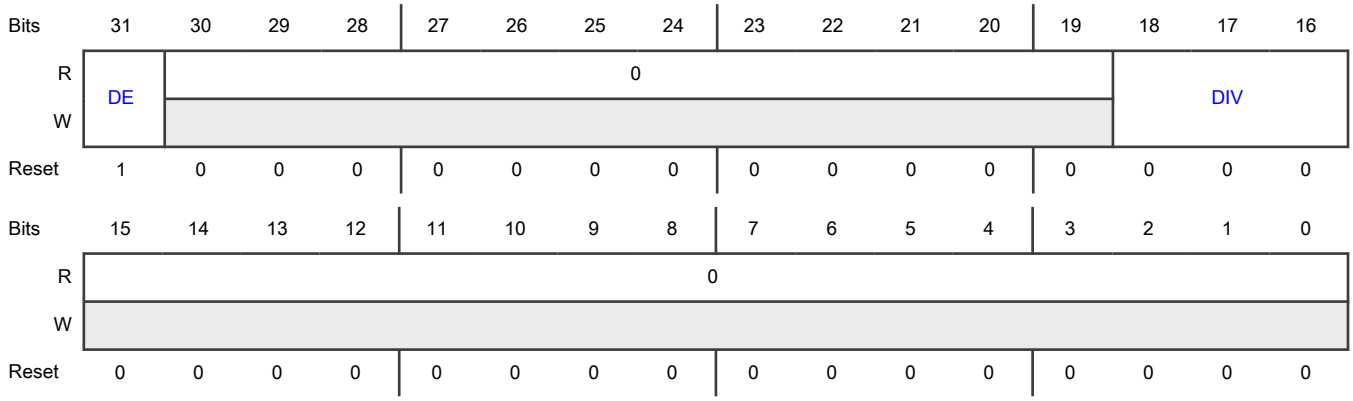
This register controls the clock divider 0 for clock mux 11.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.61 Clock Mux 11 Divider Update Status Register (MUX_11_DIV_UPD_STAT)

Offset

Register	Offset
MUX_11_DIV_UPD_STAT	5FCh

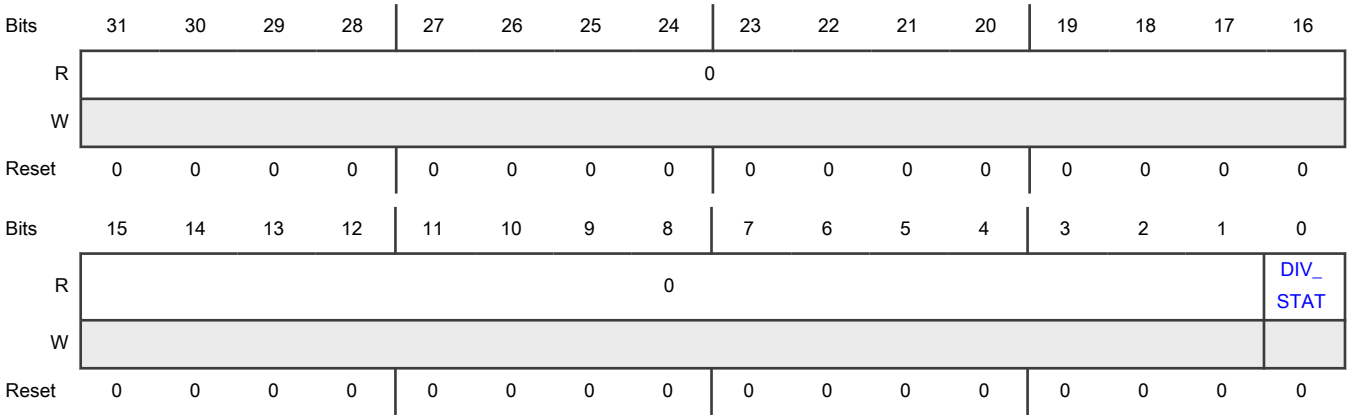
Function

This register provides the update status of the clock dividers corresponding to clock mux 11. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 11 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.6 Glossary

PCFS	Progressive clock frequency switching
LCM	Least common multiple

Chapter 25
Fast Internal RC Oscillator (FIRC)

25.1 Overview

The FIRC digital interface controls the internal 48 MHz RC oscillator system.

25.1.1 Features

FIRC can be disabled in Standby mode via software.

- Status register provides the current operating state:
  - On and stable
  - Off or on but not stable

25.2 External signals

This module has no external signals.

25.3 Initialization

This module does not require initialization.

25.4 FIRC register descriptions

25.4.1 FIRC memory map

FIRC base address: 402D_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	Status Register (Status_Register)	32	R	0000_0001h
8h	Standby Enable Register (STDBY_ENABLE)	32	RW	0000_0000h

25.4.2 Status Register (Status_Register)

Offset

Register	Offset
Status_Register	4h

Diagram

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

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

Fields

Field	Function
31-1 —	Reserved
0 STATUS	Status bit for FIRC 0b - FIRC is off or unstable. 1b - FIRC is on and stable.

25.4.3 Standby Enable Register (STDBY_ENABLE)

Offset

Register	Offset
STDBY_ENABLE	8h

Function

This register enables or disables FIRC in chip's Standby mode.

Diagram

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

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

Fields

Field	Function
31-1 —	RESERVED
0 STDBY_EN	Enables or disables FIRC in chip's Standby mode. 0b - Disabled 1b - Enabled

25.5 Glossary

Standby mode	Power saving mode of the chip

Chapter 26
Slow Internal RC Oscillator (SIRC)

26.1 Introduction

The SIRC digital interface controls the slow internal on-chip 32 KHz RC oscillator system.

26.2 Features

The SIRC module:

- Status register provides the current operating state:
  - On and stable
  - Off or on but not stable
- Operates at a frequency of 32 kHz in Functional mode

26.3 Operating mode

Only a POR reset will initialize the SIRC. Destructive or Functional resets do not impact the SIRC functionality.

SIRC stabilization occurs after 96 SIRC_CLK cycles.

The SIRC output clock remains invalid until the analog SIRC stabilizes. The output clock does not glitch or overshoot its frequency during enabling or disabling. Also, the clock does not get stuck or produce glitches on a very short hardware disable pulse.

26.4 SIRC register descriptions

26.4.1 SIRC memory map

Access to registers use 8-bit, 16-bit, or 32-bit addressing.

SIRC base address: 402C_8000h

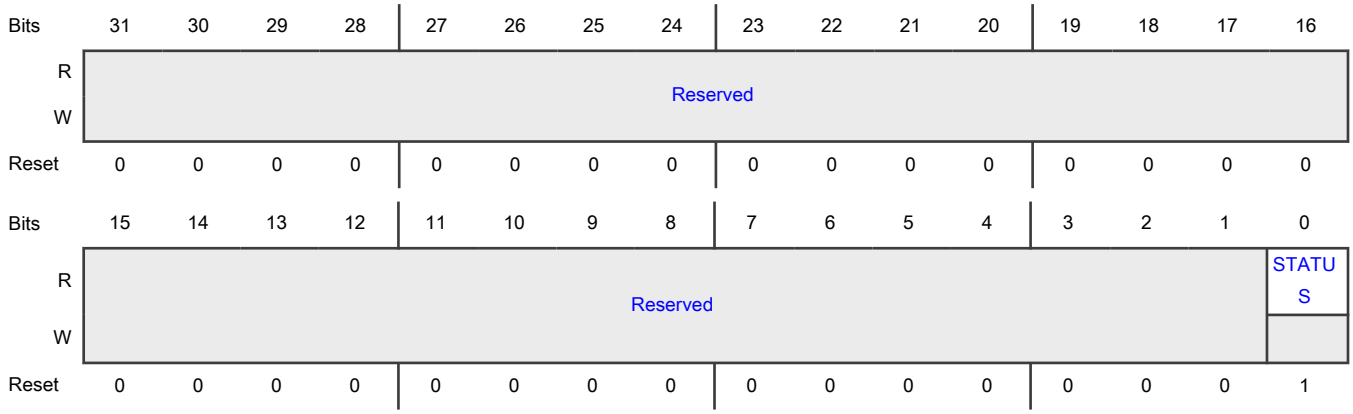
Offset	Register	Width (In bits)	Access	Reset value
4h	Status Register (SR)	32	R	0000_0001h
Ch	Miscellaneous input (MISCELLANEOUS_IN)	32	RW	0000_0000h

26.4.2 Status Register (SR)

Offset

Register	Offset
SR	4h

Diagram



Fields

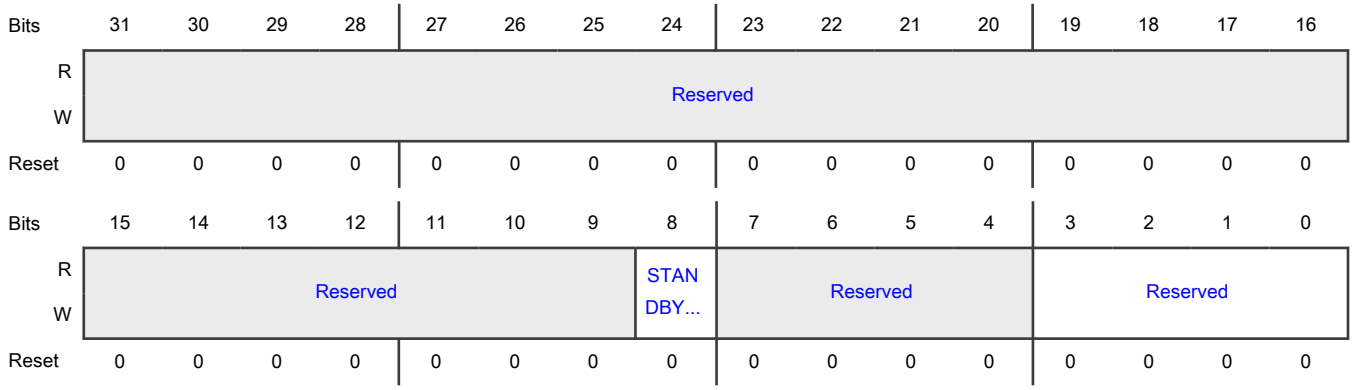
Field	Function
31-1 —	Reserved
0 STATUS	Status bit for SIRC 0b - SIRC is off or unstable 1b - SIRC is on and stable

26.4.3 Miscellaneous input (MISCELLANEOUS_IN)

Offset

Register	Offset
MISCELLANEOUS_IN	Ch

Diagram



Fields

Field	Function
31-9 —	Reserved
8 STANDBY_ENABLE	Standby Enable for SIRC 0b - SIRC disables in Standby mode 1b - SIRC enables in Standby mode
7-4 —	Reserved
3-0 —	Reserved

Chapter 27
Fast Crystal Oscillator Digital Controller (FXOSC)

27.1 Chip-specific FXOSC information

27.1.1 Chip-specific FXOSC information

For MWCT2D17S: For bypass mode applications, the EXTAL pin should be driven low when FXOSC is in off/disabled state.

- While initializing FXOSC: When the FXOSC is used in Bypass mode, the external clock source can only be enabled after the FXOSC is enabled.
- While disabling FXOSC: When the FXOSC is used in Bypass mode, the external clock source must already be inactive before disabling the FXOSC.

27.2 Overview

The Fast Crystal Oscillator (FXOSC) generates a clock which can be used at the SoC level. The FXOSC has a digital interface to control and configure the oscillator. When FXOSC is powered down at any time, it is designed not to generate any glitch at the output clock. A counter inside FXOSC handles different stabilization times. CG cell is clock gating cell, it gates the clock till stabilization time.

27.2.1 Block diagram

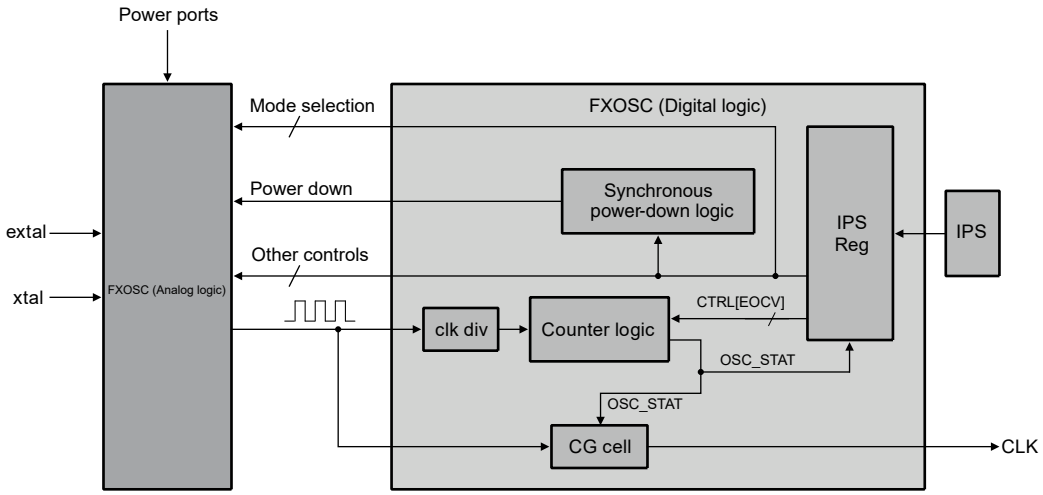


Figure 114. FXOSC block diagram

27.2.2 Features

FXOSC features are as follows:

- Status register shows current module state.
- Control register can:
  - Select a mode of operation:
    - Crystal mode
    - Single-Input Bypass mode using EXTAL clock input

— Disable the module (Power-Down mode)

27.3 Functional description

The table below shows configuration settings for the different FXOSC modes set by the CTRL register:

Table 139. FXOSC operation mode settings

Mode	CTRL[OSCON]	CTRL[OSC_BYP]	CTRL[COMP_EN]	FXOSC_CLK
Power-Down mode	0	X	X	0
Crystal mode	1	0	1	Crystal clock
Single-Input Bypass mode	1	1	0	EXTAL

Power-Down mode is the FXOSC default condition after any reset: POR, Destructive, or Functional.

27.3.1 Clock generation in Crystal mode

FXOSC_CLK start when counter value reaches FXOSC_CTRL[EOCV] × 128.

27.3.2 Clock generation in Single-Input Bypass mode

This mode bypasses the oscillator and uses a single-input external clock (EXTAL input) for FXOSC_CLK. See the chip-specific FXOSC information for Bypass mode applications.

27.3.3 Clocking

This module has no clocking considerations.

27.3.4 Interrupts

This module has no interrupts.

27.4 External signals

This module has no external signals.

27.5 Initialization

Initializing FXOSC

Initialize FXOSC as follows:

1. Write the desired value to CTRL[OSC_BYP] and CTRL[COMP_EN] to select an operation mode as shown in Table 139.

   NOTE

   FXOSC must be disabled when the operation mode is modified.

2. Configure CTRL[GM_SEL].
   - In Crystal mode configure the transconductance based on the module specification in the chip data sheet.
   - In Single-Input Bypass mode write 0000b to this field.

   NOTE

   In Crystal mode FXOSC will not function with zero transconductance (GM_SEL = 0000b).

3. Set CTRL[EOCV] calculating the value as follows:
   - EOCV (in decimal) = (stabilization time in ns) ÷ (4×128×(period of clock in ns))
4. Write 1 to CRTL[OSCON] to enable FXOSC.

NOTE

When the FXOSC is used in Bypass mode, the clock from the external source can only be used after the FXOSC is enabled

5. Confirm the clock is stable (STAT[OSC_STAT] = 1) before using it.

NOTE

See Hardware design guide for further details and the recommended circuit for each mode.

Disabling FXOSC

Write 0 to CRTL[OSCON] to disable FXOSC when FXOSC_CLK is running and stable.

FXOSC enters Power-down mode after at least four crystal clocks. No glitches occur during the transition to Power-Down mode because synchronizers are used.

NOTE

After disabling FXOSC:

- Wait for at least 2µs before enabling FXOSC again.
- You must not change other values in FXOSC registers for at least 16 FXOSC_CLK cycles.

27.6 FXOSC register descriptions

This section provides the descriptions of all registers used for configuring the FXOSC.

27.6.1 FXOSC memory map

Use 8-bit, 16-bit, or 32-bit addressing to access registers.

FXOSC base address: 402D_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	FXOSC Control Register (CTRL)	32	RW	019D_00C0h
4h	Oscillator Status Register (STAT)	32	R	0000_0000h

27.6.2 FXOSC Control Register (CTRL)

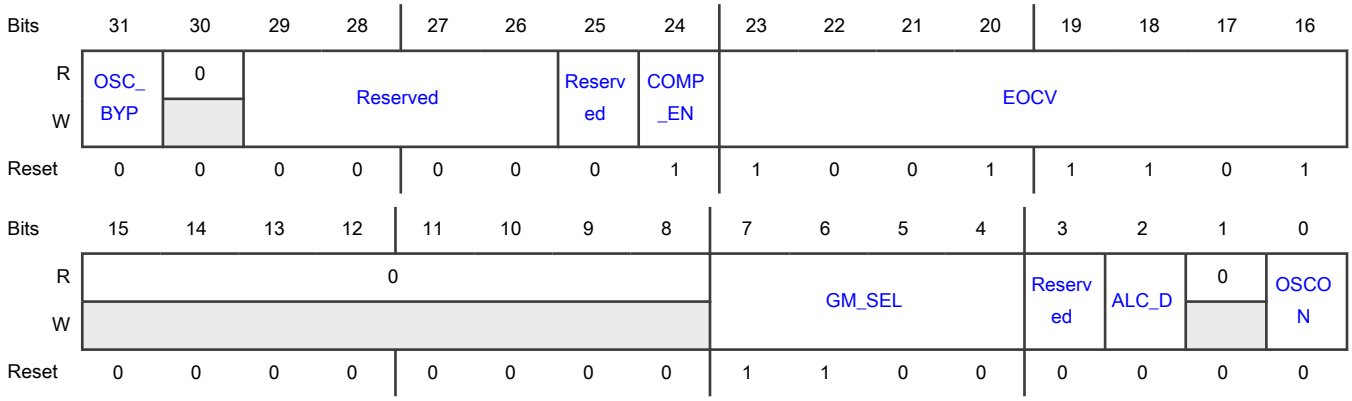
Offset

Register	Offset
CTRL	0h

Function

Configures FXOSC operation.

Diagram



Fields

Field	Function
31 OSC_BYP	Oscillator bypass Bypasses the internal oscillator. 0b - Internal oscillator not bypassed 1b - Internal oscillator bypassed
30 —	Reserved
29-26 —	Reserved
25 —	Reserved
24 COMP_EN	Comparator enable Enables or disables the comparator. • For Crystal mode set this field to 1. • For Single-Input Bypass mode set this field to 0. 0b - Comparator disabled 1b - Comparator enabled
23-16 EOCV	End of count value Specifies the end-of-count. The oscillator counter runs on the crystal clock divided by 4 and counts up to EOCV × 128. This counting period ensures that the external oscillator clock signal is stable before the system selects FXOSC as a source.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE • You must set EOCV to the appropriate value to allow clock and duty cycle to stabilize and guarantee that OSC_STAT becomes set within the crystal startup time. — In Crystal mode, EOCV value must to be calculated to appropriate value based on the crystal specification using the equation in Initializing FXOSC. — In Single-Input Bypass mode, EOCV value is irrelevant. FXOSC holds the counter in reset. • Before modifying EOCV, FXOSC must be disabled.
15-8 —	Reserved
7-4 GM_SEL	Crystal overdrive protection Selects the transconductance applied by the FXOSC amplifier. This setting depends on crystal specification. **NOTE** • In Crystal mode FXOSC will not function with zero transconductance (GM_SEL = 0000b). • For details on how to set this field, see Initializing FXOSC. 0000b - 0x 0001b - 0.1004x 0010b - 0.2009x 0011b - 0.3013x 0100b - 0.2343x 0101b - 0.3348x 0110b - 0.4345x 0111b - 0.5349x 1000b - 0.4679x 1001b - 0.5684x 1010b - 0.6681x 1011b - 0.7678x 1100b - 0.7016x 1101b - 0.8013x 1110b - 0.9003x 1111b - 1x
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 ALC_D	Automatic level controller enable The ALC feature of the FXOSC internal circuit automatically adjusts the bias current of the crystal oscillator's amplifier. It must be used in crystal oscillator mode. During startup the EXTAL/XTAL amplitude would be lower, so the ALC circuit dynamically increases the bias current of the internal amplifier to provide higher current and make crystal oscillator startup faster in normal oscillator mode. After EXTAL/XTAL amplitude has increased to a steady state, the ALC circuit automatically reduces the amplifier's bias current to save power. 0b - Enables automatic level controller 1b - Disables automatic level controller
1 —	Reserved
0 OSCON	Crystal oscillator power-down control Enables or disables FXOSC 0b - Disables FXOSC 1b - Enables FXOSC

27.6.3 Oscillator Status Register (STAT)

Offset

Register	Offset
STAT	4h

Function

Shows current state of FXOSC.

Diagram

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

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

Fields

Field	Function
31 OSC_STAT	Crystal oscillator status Indicates the crystal oscillator status. **NOTE** OSC_STAT value is not valid if transconductance is set to 0. 0b - Crystal oscillator is off or on but not stable. 1b - Crystal oscillator is on and providing a stable clock.
30-0 —	Reserved

Chapter 28
Slow Crystal Oscillator Digital Controller (SXOSC)

28.1 Introduction

SXOSC:

- Controls the functioning of the analog oscillator
- Provides a register interface for programmable features
- Controls the power-down function of the oscillator

28.2 Features

- Generates a 32 KHz clock output in Functional Oscillator mode
- Contains a status register, the value of which becomes 1 when the crystal stabilization time is complete
- Contains an oscillator that can be powered down

28.3 Modes of operation

- Functional Oscillator—In this mode, the external crystal generates the clock.

28.4 Functional description

SXOSC generates control signals to configure the analog module to operate in specific modes.

The following table shows the mode of operation available for selection and its settings.

Table 140. Operation mode settings

Mode	Value of SXOSC_CTRL[OSCON]	Output clock
Functional Oscillator	0 (oscillator switched off)	0 (indicates no output)
	1 (oscillator switched on)	Crystal clock

28.4.1 Clock generation in Functional Oscillator mode

After hard reset, the crystal oscillator is switched off by default. For clock generation in Functional Oscillator mode, see Table 140. The counter logic starts counting and the stable clock starts running one clock cycle after reaching the value of SXOSC_CTRL[EOCV] x 128 counter value. The module writes 1 to SXOSC_STAT[OSC_STAT] after two module clock cycles.

28.4.2 Clock stopping in Functional Oscillator mode

To stop a stable, running clock, configure the mode as specified in Table 140. A glitch does not occur because synchronizers are used.

28.5 Initialization information

Perform the following steps to initialize SXOSC:

1. Write an appropriate value to SXOSC_CTRL[EOCV].
2. Write 1 to SXOSC_CTRL[OSCON]. This enables the oscillator. The value of this field defaults to 0, which means the oscillator is off by default.
3. Confirm the clock is stable (SXOSC_STAT[OSC_STAT] = 1) before using it.

Be aware that:

- A functional reset does not affect any mode settings.
- Functional Oscillator is the only available mode of operation.
- To calculate EOCV of the crystal from start-up time, use the following formula:

  EOCV = (stabilisation time in ns) ÷ (4 × 128 × time period of clock in ns).

28.6 SXOSC register descriptions

This section provides the description of registers that are used for configuring SXOSC.

28.6.1 SXOSC memory map

Addresses are provided as offsets from the module base address. You can access all the registers using 8-bit, 16-bit, or 32-bit addressing.

Using external configuration signals or parameters, you can configure some of the register reset values according to the requirements of individual chips.

SXOSC base address: 402C_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Oscillator Control (SXOSC_CTRL)	32	RW	007D_0000h
4h	Oscillator Status (SXOSC_STAT)	32	R	0000_0000h

28.6.2 Oscillator Control (SXOSC_CTRL)

Offset

Register	Offset
SXOSC_CTRL	0h

Diagram

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

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

Fields

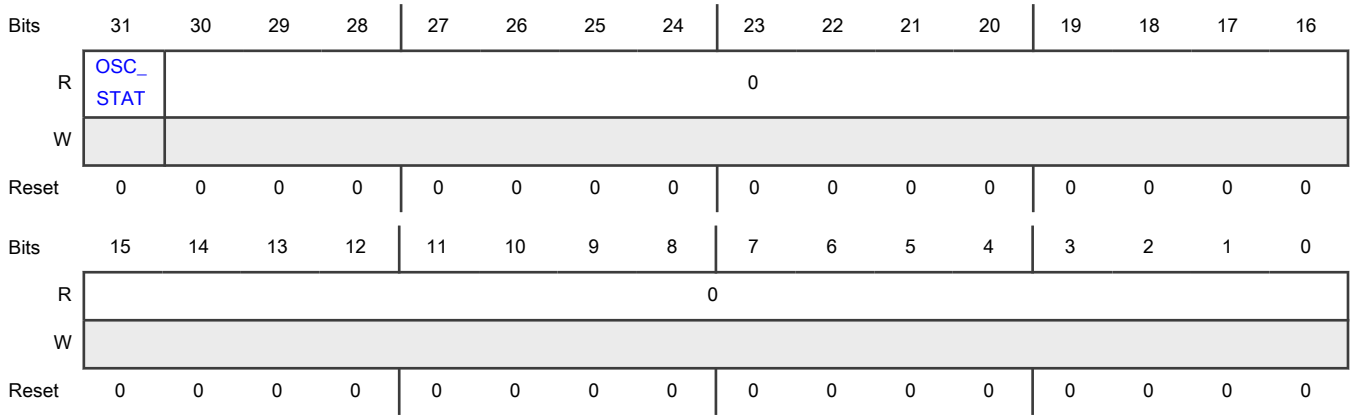
Field	Function
31-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-16 EOCV	End Of Count Value Specifies the end of count value, which the oscillator stabilization counter uses for comparison whenever the oscillator is switched on. This counting period ensures that SXOSC's clock signal is stable before the system selects it. The oscillator counter runs on a divided crystal clock (divide by 4) and counts up to 128 times the EOCV value (EOCV × 128). **NOTE** To calculate an appropriate EOCV value, the internal counter must run for at least the stabilization time specified for the crystal in the data sheet. Also, it is recommended to change the value of EOCV only when SXOSC is in the disabled state.
15-8 —	Reserved
7-6 —	Reserved
5-1 —	Reserved
0 OSCON	Crystal Oscillator Power-Down Control Indicates the status of the crystal oscillator. When disabling SXOSC, write 0 to this field and do not change other values in this register for at least 16 SXOSC clock cycles. 0b - Disabled (switched off) 1b - Enabled (switched on)

28.6.3 Oscillator Status (SXOSC_STAT)

Offset

Register	Offset
SXOSC_STAT	4h

Diagram



Fields

Field	Function
31 OSC_STAT	Crystal Oscillator Status Indicates the output clock status of the crystal oscillator. 0b - Unstable 1b - Stable
30-0 —	Reserved

Chapter 29
PLL Digital Interface (PLLDIG)

29.1 Chip-specific PLLDIG information

29.1.1 PLLDIG instances

This chip supports up to one instances of PLLDIG.

Table 141. PLLDIG instances

Instance	MWCT2014S/MWCT2015S/MWCT2016S/MWCT2D16S/MWCT2D17S
PLL	Yes

Table 142. PLLDIG configuration

Instance	Frequency modulation supported	Number of reference clocks supported	Number of clock outputs supported
PLL	Yes	1[1]	2

1. For MWCT2015S: 2

29.1.2 PLL-supported accesses and frequencies

The PLLODIV_0 and PLLODIV_1 registers support only word accesses. When you write to these registers, you must retain the default values of the reserved fields.

PLLDIG supports a down-spread modulation of up to 160 MHz PLL PHI clock output only.

29.2 Introduction

PLL can multiply or divide the frequency of a given clock input.

29.2.1 Features

PLL includes the following features:

- Programmable frequency modulation
- Multiple integer dividers on PLL outputs
- Lock detection circuitry reports when PLL achieves frequency lock
- Continuous monitoring of lock status to report Loss of Lock (LOL) condition
- Powering down the module for low-power operation (Power-Down mode)

29.2.2 Block diagram

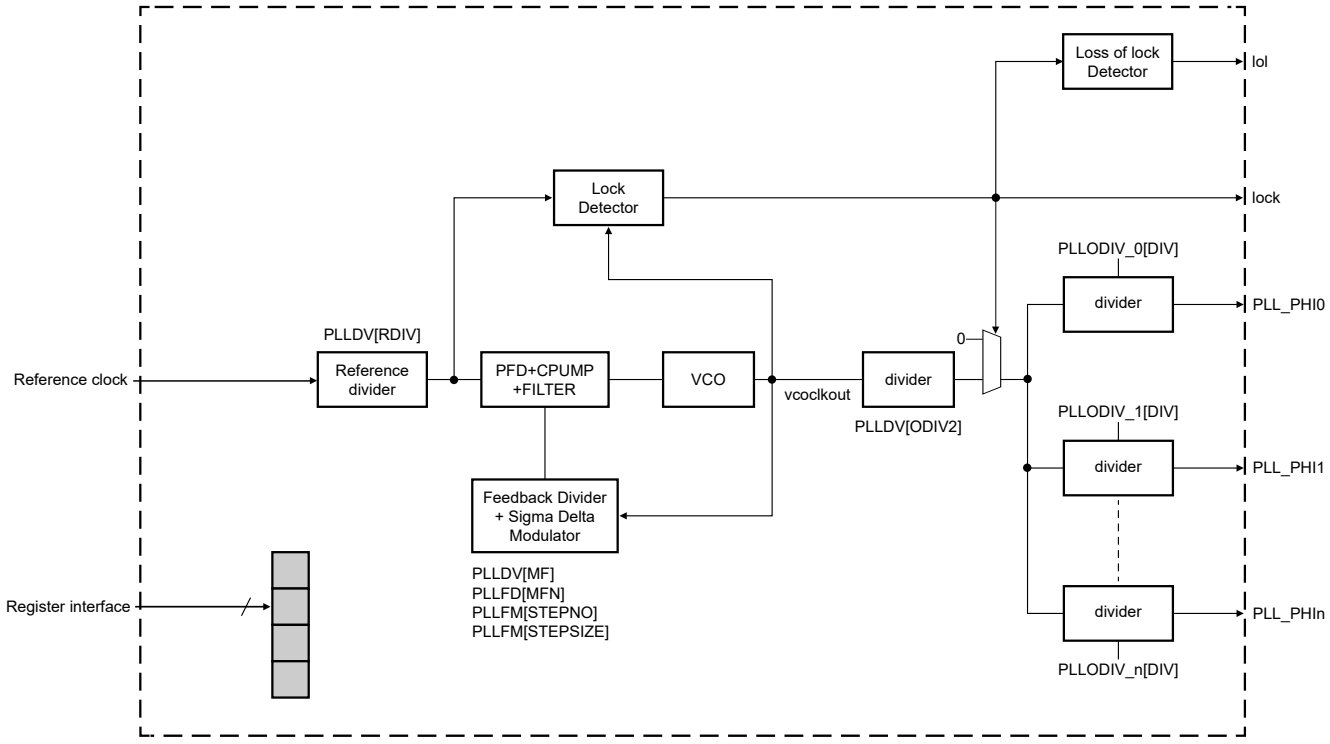


Figure 115. Block diagram

The number of output dividers can vary with the module instance. See the Clocking chapter to confirm the number of PLL output dividers.

29.3 Modes of operation

Table 143. Modes of operation

PLLCR[PLLPD]	PLLFD[SDMEN]	PLLFM[SSCGBYP]	Description
1	x	x	PLL is disabled.
0	0	1	Functional mode – PLL operates in integer-only mode. See Clock configuration.
0	1	1	Functional mode – PLL operates in Fractional mode (non-Frequency modulation). See Clock configuration.
0	1	0	Functional mode – PLL operates in Frequency Modulation mode. See Frequency modulation.

29.4 Functional description

This section explains PLL operation and configuration.

29.4.1 Input clock frequency

PLL is designed to operate over a specified input clock frequency range. PLL source frequency limits are discussed in this chip's data sheet.

29.4.2 Clock configuration

See the equations below and the corresponding register configuration that determine the relationship between VCO frequency (f_{VCO}) and PLL reference frequency.

- Integer-only mode:
  - When PLLDV[RDIV] is 0:

$$f_{\text{pll_VCO}} = f\text{pll_ref} \times \text{PLLDV[MFI]}$$

Equation 1. PLL VCO frequency in integer-only mode when PLLDV[RDIV] is 0

  - When PLLDV[RDIV] is not 0:

$$f_{\text{pll_VCO}} = \frac{f_{\text{pll_ref}}}{\text{PLLDV[RDIV]}} \times \text{PLLDV[MFI]}$$

Equation 2. PLL VCO frequency in integer-only mode when PLLDV[RDIV] is not 0

- Fractional mode:
  - When PLLDV[RDIV] is 0:

$$f_{\text{pll_VCO}} = f_{\text{pll_ref}} \times \left(\text{PLLDV[MFI]} + \frac{\text{PLLFD[MFN]}}{18432}\right)$$

Equation 3. PLL VCO frequency in Fractional mode when PLLDV[RDIV] is 0

  - When PLLDV[RDIV] is not 0:

$$f_{\text{pll_VCO}} = \frac{f_{\text{pll_ref}}}{\text{PLLDV[RDIV]}} \times \left(\text{PLLDV[MFI]} + \frac{\text{PLLFD[MFN]}}{18432}\right)$$

Equation 4. PLL VCO frequency in Fractional mode when PLLDV[RDIV] is not 0

See the equation below and the corresponding register configuration that determine the relationship between reference and PLL_PHI*n* output frequencies.

$$f_{\text{pll_phi}} = \frac{f_{\text{pll_VCO}}}{\text{PLLDV[ODIV2]} \times \left(\text{PLLDIV_}n\text{[DIV]} + 1\right)}$$

Equation 5. PLL PHI output frequency

When configuring PLL, you must not violate the maximum system clock frequency or maximum and minimum VCO frequency specification of PLL (see this chip's data sheet for frequency limits).

You must disable PLL by writing 1 to PLLCR[PLLPD] before any PLL configuration or input clock are modified.

You must disable PLL by writing 1 to PLLCR[PLLPD] for at least 5 µs before writing 0 to PLLCR[PLLPD] to enable PLL.

The recommended procedure to program PLL and enter Normal mode is shown in Initialization information.

29.4.3 LOL

PLL provides LOL indication. The LOL indication can only be generated when PLL is in Functional mode (see Modes of operation). When PLL detects a LOL, it asserts its LOL event output.

PLL does not detect loss of reference clock. If the reference clock stops after PLL achieves lock, PLL continues to indicate lock. It is assumed that monitoring of PLL's reference clock that is done outside PLL is enabled while PLL is in operation.

PLL LOL is intended for detection of gross failures. Use CMUs for accurate frequency monitoring.

29.4.4 Frequency modulation

In Frequency Modulation mode, PLL generates a frequency-modulated clock. The modulation depth and modulation frequency are calculated using the equations shown in Frequency modulation programming.

Write 1 to PLLFM[SPREADCTL] to select down-spread modulation. See Figure 116 that shows an example of down-spread modulation.

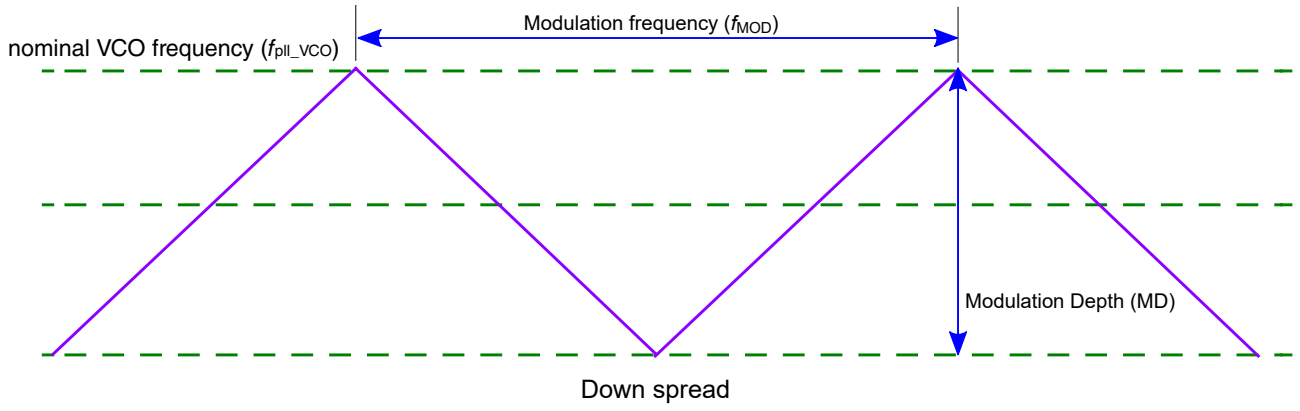


Figure 116. Frequency modulation

29.4.4.1 Frequency modulation programming

Modulation depth and modulation frequency programming uses step number (PLLFM[STEPNO]) and step size (PLLFM[STEPSIZE]). The table below shows variables used during calculations when programming PLL for frequency modulation.

Table 144. Variables for configuring modulation depth and frequency

Variable	Description
f_{REF}	Input clock frequency
f_{MOD}	Expected modulation frequency
MD	Expected modulation depth in percentage
LDF	Loop division factor

Use the following equations to configure PLL for frequency modulation.

$$LDF = PLLDV[MFI] + \frac{PLLFD[MFN]}{18432}$$

Equation 6. LDF

$$PLLFM[STEPNO] = \frac{f_{REF}}{\left(2 \times f_{MOD} \times PLLDV[RDIV]\right)}$$

Equation 7. Step number

$$\text{PLLFM[STEPSIZE]} = \frac{\text{MD} \times \text{LDF}}{100 \times \text{PLLFM[STEPNO]}} \times 18432$$

Equation 8. Step size

Frequency modulation is only possible if the condition shown in Equation 9 is met.

$$(\text{PLLFM[STEPSIZE]} \times \text{PLLFM[STEPNO]}) < 18432$$

Equation 9. Requirement to achieve FM

You must write 0 to PLLFM[SSCGBYP] and write 1 to PLLFD[SDMEN] to enable frequency modulation.

CAUTION

The effective modulation depth may differ from the intended modulation depth because of rounding operations applied to PLLFM[STEPSIZE] and PLLFM[STEPNO].

29.5 Initialization information

Perform the following steps to initialize PLL:

1. Confirm that PLLODIV_*n*[DE] is 0 for all dividers.
2. Confirm that PLLCR[PLLPD] is 1.
3. Program the following as needed:
   - PLLDV
   - PLLFD
   - PLLFM to the desired value
4. Program PLLDV[ODIV2] and PLLODIV_*n*[DIV] to the desired values.
5. Wait for the PLL reference clock to be stable.
6. Write 0 to PLLCR[PLLPD].
7. Wait for PLLSR[LOCK] to be 1.
8. Write 1 to PLLODIV_*n*[DE].

Perform the following steps to shut down PLL:

1. Write 0 to PLLODIV_n[DE] for all dividers.
2. Write 1 to PLLCR[PLLPD].

29.6 PLLDIG register descriptions

This section provides the memory map and detailed descriptions of registers used for configuring PLL. The table below shows the memory map. Addresses are given as offsets from the module base address. All registers are accessed using 8-bit, 16-bit, or 32-bit addressing.

29.6.1 PLLDIG memory map

PLL base address: 402E_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PLL Control (PLLCR)	32	RW	8000_0000h
4h	PLL Status (PLLSR)	32	RW	0000_0300h
8h	PLL Divider (PLLDV)	32	RW	0C3F_1032h
Ch	PLL Frequency Modulation (PLLFM)	32	RW	4000_0000h
10h	PLL Fractional Divider (PLLFD)	32	RW	0000_0000h
18h	PLL Calibration Register 2 (PLLCAL2)	32	RW	0006_0000h
80h - 84h	PLL Output Divider (PLLODIV_0 - PLLODIV_1)	32	RW	0000_0000h

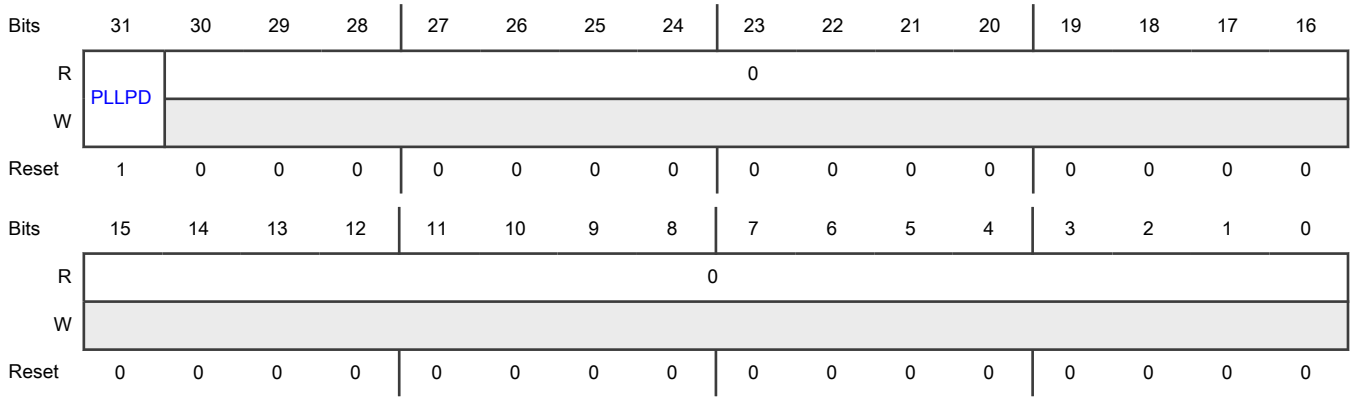
29.6.2 PLL Control (PLLCR)

Offset

Register	Offset
PLLCR	0h

Function
Configures PLL functionality.

Diagram



Fields

Field	Function
31 PLLPD	PLL Power Down Powers down or powers up PLL. 0b - Powered up

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Powered down
30-0 —	Reserved

29.6.3 PLL Status (PLLSR)

Offset

Register	Offset
PLLSR	4h

Function
Shows the PLL status.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				Reserved				0				LOL	LOCK	0	
W													W1C			
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-11 —	Reserved
10-8 —	Reserved
7-4 —	Reserved
3	Loss-Of-Lock Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
LOL	Indicates the current PLL lock status. 0b - No loss of lock detected 1b - Loss of lock detected
2 LOCK	Lock Status Indicates that PLL has acquired lock. 0b - Unlocked 1b - Locked
1-0 —	Reserved

29.6.4 PLL Divider (PLLDV)

Offset

Register	Offset
PLLDV	8h

Function

Divides input clocks for PLL output generation.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	ODIV2							0				Reserved			
W																
Reset	0	0	0	0	1	1	0	0	0	0	1	1	1	1	1	1

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

Fields

Field	Function
31	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
30-25 ODIV2	Output frequency divider for raw PLL clock. 6-bit field determining the VCO clock post divider for driving the PHI output clock. 000000 – Divide by 1 000001 – Divide by 1 000010 – Divide by 2 000011 – Divide by 3 000100 – Divide by 4 ..... 111111 – Divide by 63
24-22 —	Reserved
21-16 —	Reserved
15 —	Reserved
14-12 RDIV	Input Clock Predivider Sets the input clock divider. The output of the predivider circuit generates the PLL loop reference clock. 000b - Divide by 1 001b - Divide by 1 010b - Divide by 2 011b - Divide by 3 100b - Divide by 4 101b - Divide by 5 110b - Divide by 6 111b - Divide by 7
11-8 —	Reserved
7-0 MFI	Integer Portion Of Loop Divider Sets the value of the divider in the PLL feedback loop.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The value specified establishes the multiplication factor applied to the reference frequency. Write the divider value to this field, where the chosen value does not violate VCO frequency specifications.

29.6.5 PLL Frequency Modulation (PLLFM)

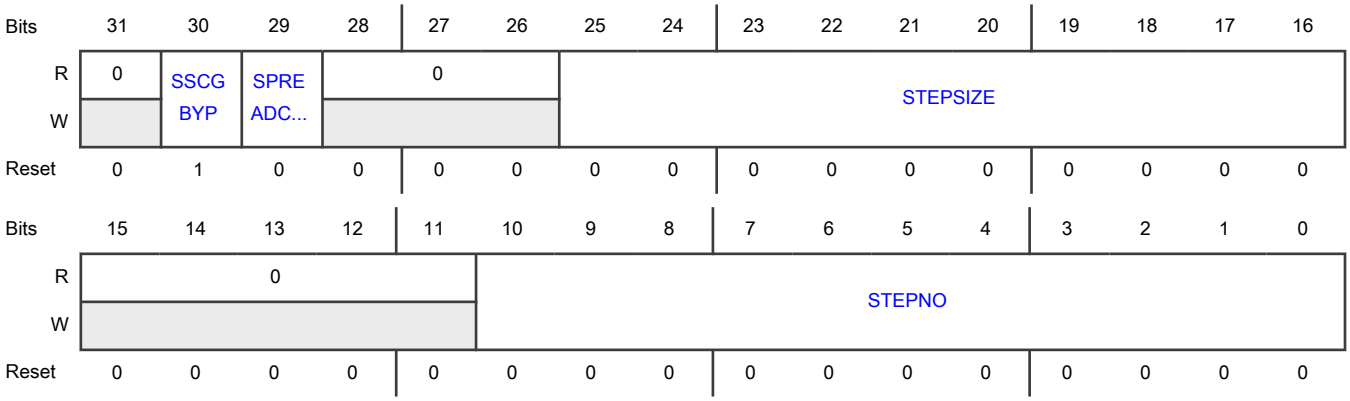
Offset

Register	Offset
PLLFM	Ch

Function

Configures PLL frequency modulation parameters.

Diagram



Fields

Field	Function
31 —	Reserved
30 SSCGBYP	Frequency Modulation (Spread Spectrum Clock Generation) Bypass Bypasses frequency modulation. 0b - Not bypassed 1b - Bypassed
29 SPREADCTL	Modulation Type Selection Indicates that the modulation is spread below the nominal frequency. You must write 1 to this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Reserved 1b - Spread below nominal frequency
28-26 —	Reserved
25-16 STEPSIZE	Frequency Modulation Step Size Provides the step size for modulation depth and frequency in Frequency Modulation mode (see Frequency modulation).
15-11 —	Reserved
10-0 STEPNO	Number Of Steps Of Modulation Period Or Frequency Modulation Provides the number of steps to achieve modulation depth in Frequency Modulation mode (see Frequency modulation).

29.6.6 PLL Fractional Divider (PLLFD)

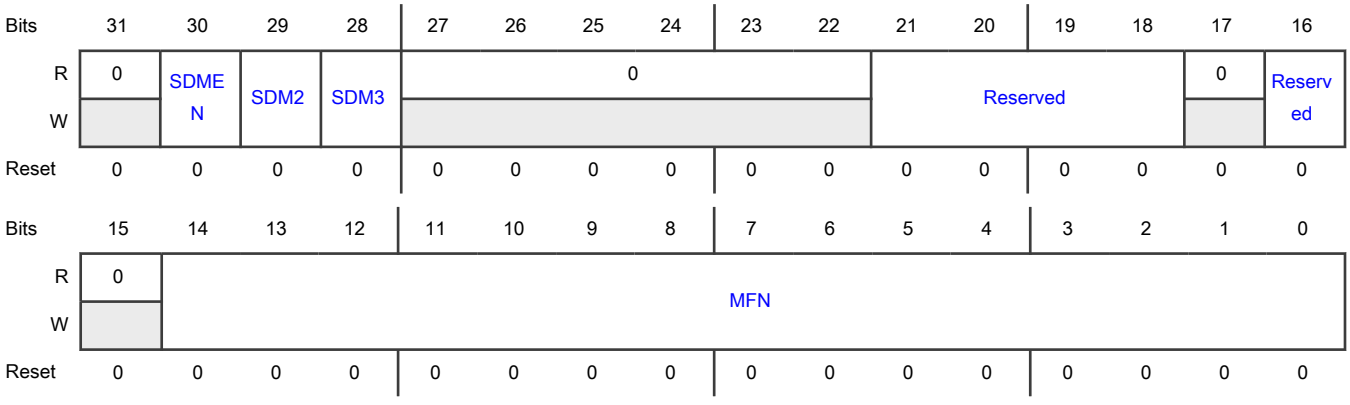
Offset

Register	Offset
PLLFD	10h

Function

Enables and configures frequency modulation.

Diagram



Fields

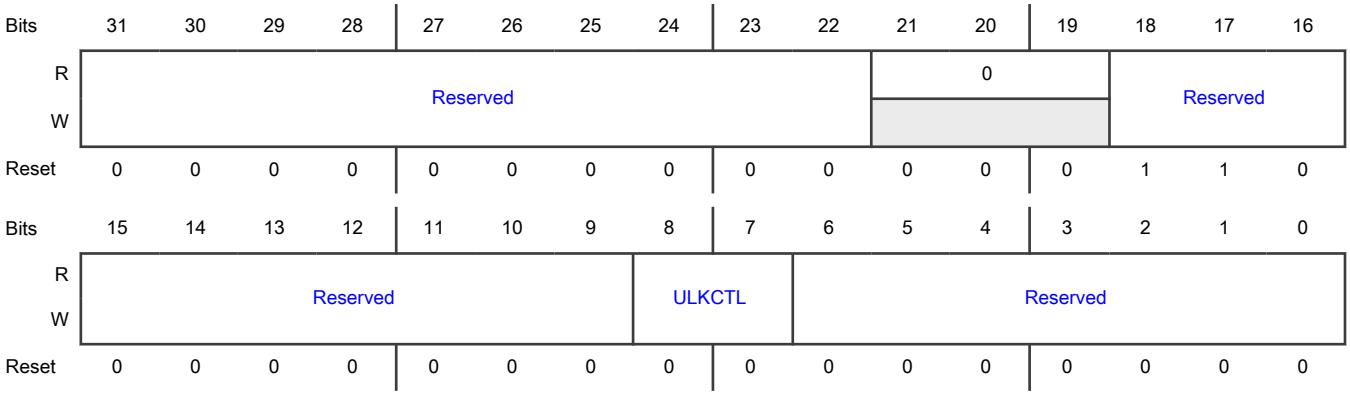
Field	Function
31 —	Reserved
30 SDMEN	Fractional Mode Enable Enables Fractional mode. 0b - Disabled 1b - Enabled
29 SDM2	Fractional Mode Configuration When you are in the fractional mode (SDMEN = 1), write 1 to this field. **NOTE** If SDMEN = 1, this field must be wrtiten 1.
28 SDM3	Fractional Mode Configuration When you are in the fractional mode (SDMEN = 1), write 1 to this field. **NOTE** If SDMEN = 1, this field must be wrtiten 1.
27-22 —	Reserved
21-18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14-0 MFN	Numerator Of Fractional Loop Division Factor Sets the numerator of the fractional loop division factor. You must write a value of less than 18432 to this field. When Fractional mode is disabled, you must write 000_0000_0000_0000b to this field.

29.6.7 PLL Calibration Register 2 (PLLCAL2)

Offset

Register	Offset
PLLCAL2	18h

Diagram



Fields

Field	Function
31-22 —	Reserved
21-19 —	Reserved
18-16 —	Reserved
15-9 —	Reserved
8-7 ULKCTL	Unlock Control Accuracy Defines the accuracy necessary to achieve unlock. The lock counter determines unlock if the number of VCO clock cycles in the window of reference cycles is outside the number of cycles defined by this field. 00b - Unlock range = Expected value ± 9 (recommended when PLLFM[SSCGBYP] = 1). Unlock range = Expected value ± 9 (recommended when PLLFM[SSCGBYP] = 1) 01b - Unlock range = Expected value ± 17 (recommended when PLLFM[SSCGBYP] = 0). Unlock range = Expected value ± 17 (recommended when PLLFM[SSCGBYP] = 0)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Unlock range = Expected value ± 33 11b - Unlock range = Expected value ± 5
6-0 —	Reserved

29.6.8 PLL Output Divider (PLLODIV_0 - PLLODIV_1)

Offset

Register	Offset
PLLODIV_0	80h
PLLODIV_1	84h

Function

Controls the PLL output clock divider settings.

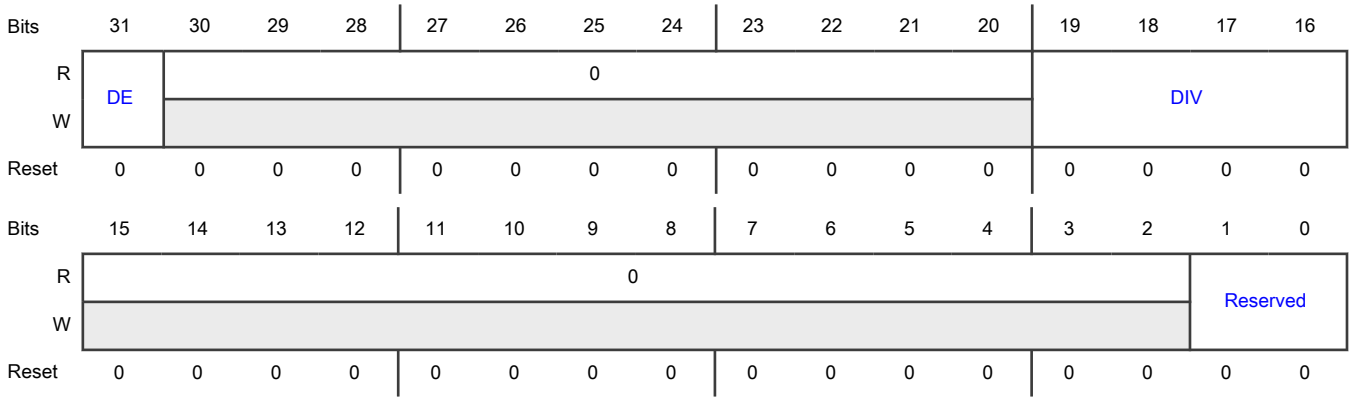
This divider has a 50% duty cycle.

NOTE

These registers support only word accesses. Other write accesses lead to the following:

- Unpredictable behavior
- No transfer error generated

Diagram



Fields

Field	Function
31 DE	Divider Enable Enables PLL output divider. Divider must be disabled before disabling PLL. 0b - Disabled 1b - Enabled
30-20 —	Reserved
19-16 DIV	Division Value Provides the division value for the output clock divider. The clock period of the clock after division is DIV + 1 times the time period of the divider input clock.
15-2 —	Reserved
1-0 —	Reserved Do not write any value other than the reset value.

Chapter 30
Reset Overview

30.1 Introduction

This chip's reset logic consists of a reset sequence that leads the chip to a fixed deterministic state after predefined reset events occur. These events can pertain to chip failure events, the chip's special operating conditions, or certain software-governed events to initiate a chip reset sequence. This chapter discusses the chip reset scheme and related topics such as:

- Types of reset reactions
- Reset event sources
- Chip reset sequences
  - POR
  - Destructive reset
  - Functional reset
- RAM retention across functional reset
- Reset pin (RESET_b) behavior
- Debug system reset
- Signal-level reset flow

30.2 Chip reset types and reactions

30.2.1 Chip reset blocks

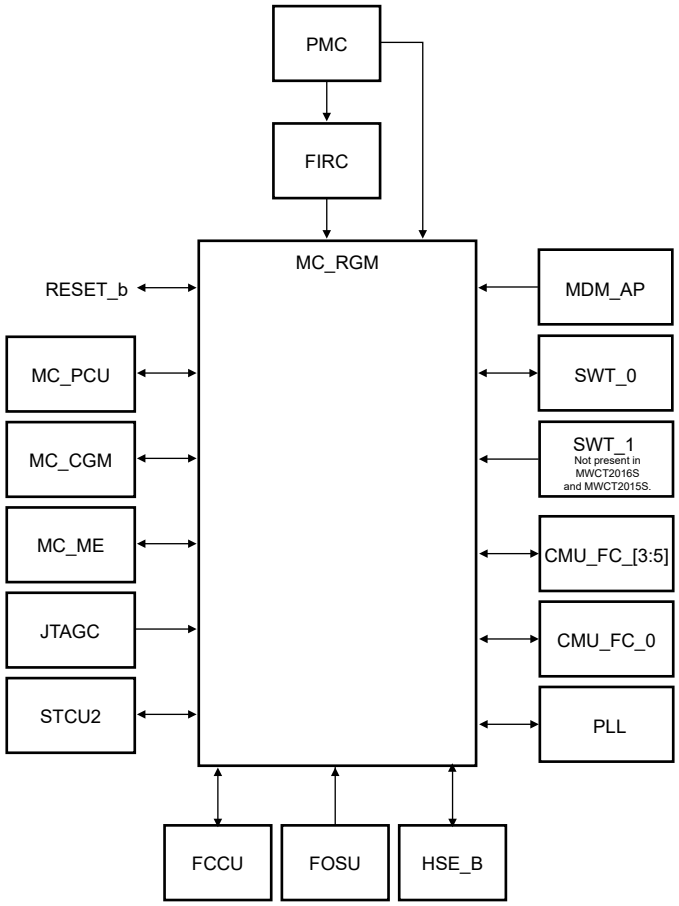


Figure 117. Chip reset blocks

30.2.2 Chip reset types

Table 145. Chip reset types

Reset event type	Functional description
POR	Leads to a complete chip reset.
Destructive	Leads most parts of the chip, except a few modules, to reset. SRAM content is lost after this reset event.
Functional	Leads all the communication peripherals and cores to reset. The communication protocols' sanity is not guaranteed and they are assumed to be reinitialized after reset. The SRAM content, and the functionality of certain modules, is preserved across functional reset.

30.2.3 High-level reset sequence overview

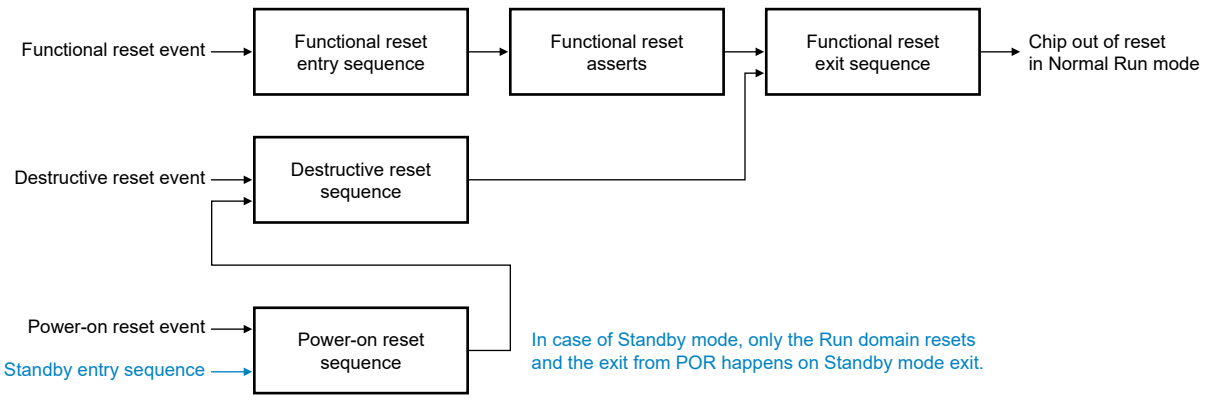


Figure 118. High-level reset sequence overview

30.2.3.1 Reset event reactions

Table 146. Reset event reactions

Reset event type	Triggered from	Reaction
POR	Anywhere	Moves to the beginning of the power-on sequence
Destructive reset	Power-on sequence	No reset sequence change
	Anywhere in the chip operation except in the power-on-sequence	Moves to the beginning of the destructive reset sequence
Functional reset	Out-of-reset	Moves to the beginning of the functional reset entry sequence
	Anywhere within the functional reset sequence	No reset sequence change

30.2.3.2 Chip action after reset event

For each reset event, immediately after MC_RGM captures it, the chip performs these actions:

1. Writes 1 to the corresponding reset event status fields in MC_RGM.DES and MC_RGM.FES (see the MC_RGM chapter for more information).
2. Places its pins in their default states (see the IOMUX file attached to this document for more information).
3. Asserts the RESET_b pin.

NOTE

After self-test completes, you can configure RESET_b assertion using MC_RGM.FES[ST_DONE].

4. Enters the reset sequence as described in Reset event reactions, depending on the current state and reset event type.

30.3 Reset sources—POR, destructive, and functional

MC_RGM records reset events in MC_RGM.FES and MC_RGM.DES, indicating the source of functional reset events and destructive reset events, respectively. You must read these fields to identify the reset source on reset recovery.

30.3.1 POR sources

Table 147. POR sources

Source module	Field in MC_RGM.DES	RESET_b assertion	Description
PMC	F_POR	Always	VDD_LV POR
			LVR on 1.1 V supply in Standby mode
			LVR on 1.1 V supply in Run mode
			LVR on 2.5 V supply in Standby mode
			LVR on 2.5 V supply in Run mode
			LVR on VDD_HV_A supply in Standby mode
			LVR on VDD_HV_A supply in Run mode
			LVR on VDD_HV_B supply in Standby mode [1]
			LVR on VDD_HV_B supply in Run mode [1]
POR_WDG			POR_WDG timeout (see the POR_WDG chapter for more information)

1. LVR on VDD_HV_B run mode and LVR on VDD_HV_B standby mode are not present in MWCT2016S and MWCT2015S.

NOTE

You cannot escalate or demote POR to an interrupt.

30.3.2 Stages of the POR sequence

Table 148. Stages of the POR sequence

Stage	Process
PWRUP	1. Starts after a POR event (for example, a POR source assert). 2. Waits for the power-up sequence to complete. 3. Exits when all the POR sources clear. 4. Transitions to the FIRC_STRT stage after the procedure completes.
FIRC_STRT	1. Enters this stage after exiting the PWRUP stage. **NOTE** FIRC_CLK, if enabled, becomes available after it is stable. The duration depends on the clock startup time (see the chip datasheet for more information). The MC_RGM state machine proceeds further after FIRC_CLK is available. 2. Transitions to the destructive reset sequence after the procedure completes.

30.3.3 Destructive reset sources

Table 149. Destructive reset sources

Source module[1]	Field in MC_RGM.DES	Description
FOSU	FCCU_FTR	FCCU failure to react
STCU2	STCU_URF	STCU2 unrecoverable fault
MC_RGM	MC_RGM_FRE	Functional reset escalation
CMU_FC_0	FXOSC_FAIL	FXOSC failure
PLL	PLL_LOL	PLL loss of lock
CMU_FC_3	CORE_CLK_FAIL	Core clock failure
CMU_FC_4	AIPS_PLAT_CLK_FAIL	AIPS_PLAT_CLK failure
CMU_FC_5	HSE_CLK_FAIL	HSE_CLK failure
CMU_FC_6	CM7_CORE CLK FAIL	CM7_CORE_CLK Failure
MC_CGM	SYS_DIV_FAIL	System clock dividers alignment failure
HSE_B	HSE_TMPR_R ST	HSE_B tamper detect reset
HSE_B	HSE_SNVS_RST	HSE_B SNVS tamper detection
MC_ME	SW_DEST	Software destructive reset
MDM_AP	DEBUG_DEST	Debug destructive reset
RESET_b pin	EXT_RST	RESET_b pin assertion

1. All destructive resets can be escalated, but only the PLL LOL destructive reset can be demoted to an interrupt (see Destructive reset event bypass for PLL LOL destructive reset bypass details).

NOTE

All reset sources in the table above assert the RESET_b pin.

30.3.4 Destructive sequence stage description

Table 150. DEST0 description

Stage	Process
DEST0	1. Asserts reset to the entire chip, except logic running on POR. 2. Waits for all the destructive reset events to clear. 3. Waits for the minimum destructive reset assertion duration of eight FIRC_CLK cycles. 4. Deasserts after stage completion.

30.3.5 Functional reset sources

Table 151. Functional reset sources

Source module	Field in MC_RGM.FES	RESET_b assertion	Demotable to IRQ[1]	Escalation[2]	Description
FCCU soft reaction[3]	FCCU_RST	Always	Yes[4]	Yes	FCCU reset reaction
STCU2	ST_DONE	Configurable	No	No	Self-test done
SWT_0	SWT0_RST	Always	Yes[5]	Yes	SWT reset request
SWT_1[6]	SWT1_RST	Always	Yes[7]	Yes	SWT reset request
JTAGC	JTAG_RST	Always	Yes [8]	No	JTAG reset
HSE_B	HSE_SWT_RST	Always	No	Yes	HSE_B SWT timeout
HSE_B	HSE_BOOT_RST	Always	No	Yes	HSE_B boot reset
MC_ME	SW_FUNC	Always	No	Yes	Software functional reset
MDM_AP	DEBUG_FUNC	Always	Yes [9]	Yes	Debug functional reset

1. See Functional reset demotion to an interrupt for more information.
2. See Reset escalation for more information.
3. An FCCU soft functional reset is a chip functional reset (see the FCCU chapter for more information).
4. Controlled by MC_RGM.FERD[D_FCCU_RST].
5. Controlled by MC_RGM.FERD[D_SWT0_RST].
6. SWT_1 is not present in MWCT2016S and MWCT2015S.
7. Controlled by MC_RGM.FERD[D_SWT1_RST].
8. Controlled by MC_RGM.FERD[D_JTAG_RST].
9. Controlled by MC_RGM.FERD[D_DEBUG_FUNC].

30.3.6 Functional reset sequence descriptions

Table 152. Functional reset sequence descriptions

Stage	Series of events
Functional reset entry sequences	
FUNC0	This stage starts after any functional reset event. The FCCU fault monitoring and CMU_F*x_n* monitoring for FLL events is masked in this step to avoid any false fault or reset.
FUNC1	In this stage, a halt sequence that includes daisy chaining of all the gaskets halts, disabling the crossbar. The stage completes after the halt-handshake sequence completes.
FUNC2	In this stage, MC_RGM triggers the MC_CGM hardware clock multiplexers to switch to FIRC_CLK. • Software-based clock multiplexers do not support switching to FIRC_CLK on functional reset. • If PCFS is enabled, the system clock switching can be done via PCFS. This stage completes after MC_CGM switches the system clock to FIRC.

Table continues on the next page...

Table 152. Functional reset sequence descriptions (continued)

Stage	Series of events
FUNC3	In this stage, MC_RGM triggers all the MC_CGM hardware-based clock multiplexers with PCFS enabled or disabled to move their dividers to default values. • Software-based clock multiplexers do not support this feature. This stage completes after all the clock multiplexer dividers initialize to their corresponding default values.
FUNC4	In this stage, PLLDIG turns off synchronously. The stage completes after PLLDIG turns off.
FUNC5	FXOSC_CLK switches off synchronously. The FUNC4 and FUNC5 stages ensure that PLLDIG disables cleanly to ensure there are no glitches on the PLLDIG clock because of reset. The stage completes after FXOSC switches off.
FUNC6	In this stage, clocks of modules that are a part of LBIST and working on the destructive reset are enabled to meet their synchronous reset requirements, if any. For the self-test logic, in self-test, the destructive reset deasserts after this stage completes and after safe stating is removed. **NOTE** In the self-test sequence, the logic, which is a part of self-test (LBIST logic) resets when self-test completes. All the parts of logic in self-test (POR, destructive, and functional reset) reset after self-test completes whereas the rest of the chip undergoes a functional reset sequence because of self-test completion (MC_RGM.FES[ST_DONE] = 1). See the Safety Overview and STCU2 chapters for more information on the self-test operation.
FUNC7	MC_RGM asserts the functional reset and triggers a counter running on FIRC_CLK (for up to 64 cycles) to enable clocks for the modules having synchronous reset requirements. Flash memory comes out of reset after this stage completes. **NOTE** The flash memory resets after a functional reset event but comes out of reset, before the rest of the modules do, at the start of the functional reset exit sequence. The rest of the modules reset when the functional reset comes out of reset at the end of the functional reset exit sequence. Therefore, the reset to flash memory is an early functional reset that deasserts earlier than the functional reset, even if asserted at the same time.
Functional reset exit sequences	
FUNC8	This stage consists of flash memory and MC_RGM handshaking. Flash memory indicates the completion of its initialization to MC_RGM.
FUNC9	DCM initiates the scanning of flash memory DCF records. This state completes after the flash memory scanning completes. See the DCF clients file attached to this document for more information.
FUNC10	After DCM scans the DCF records from the flash memory, DCM initiates the trim loading sequence for analog blocks.

Table continues on the next page...

Table 152. Functional reset sequence descriptions (continued)

Stage	Series of events
	The analog blocks are loaded with the configured trimmed values in this stage, which completes after a trim-loading sequence completes.
FUNC11	In this stage, MG_RGM stops driving RESET_b and checks that the signal does not assert externally. If you enable low-power debug, MC_RGM waits for a debug acknowledge. The completion of this stage indicates that MC_RGM completed the reset sequence, deasserting the functional reset to the system.

30.4 Reset and boot sequence

The chip reset sequence consists of several reset stages based on the occurrence of a particular reset event. All reset events follow the same chip reset sequence; only the entry points vary depending on the type of reset event. MC_RGM triggers each stage after the previous stage completes. These stages execute in a specific order, which ensures a deterministic state of the chip when a reset event completes.

Figure 119 shows a high-level representation of the chip startup sequence.

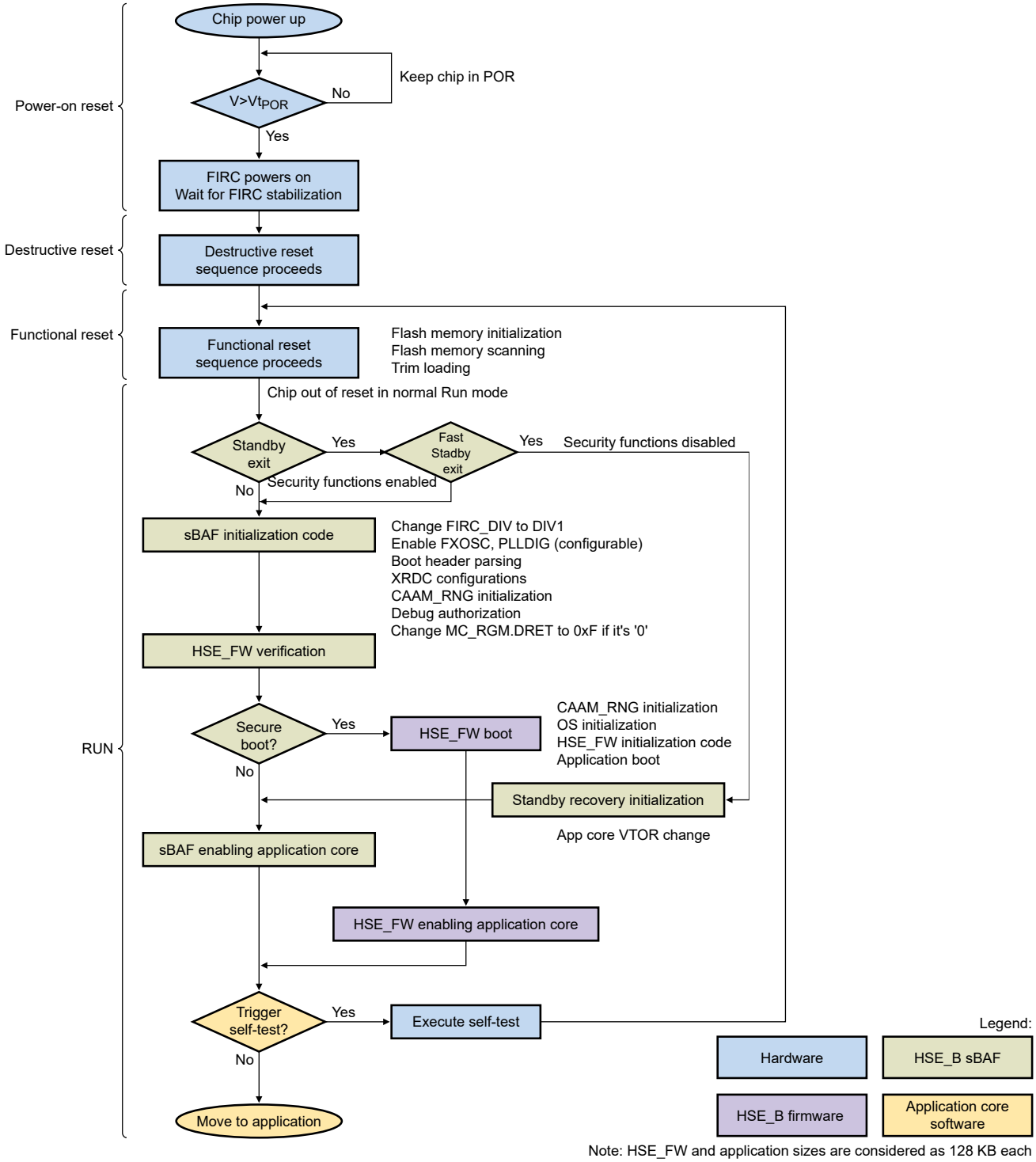


Figure 119. Chip reset and boot overview

30.4.1 POR

This stage starts when the POR event occurs, that is, when a POR source asserts. The logic within the Run power domain running on POR also resets in the chip Standby entry sequence.

NOTE

The logic in the Standby power domain does not reset in the chip Standby mode entry sequence.

The POR sequence consist of two stages:

1. Power-up (PWRUP)
2. FIRC oscillator start (FIRC_STRT)

See POR sequence for more information and Stages of the POR sequence for POR stages and their descriptions.

30.4.1.1 POR sequence

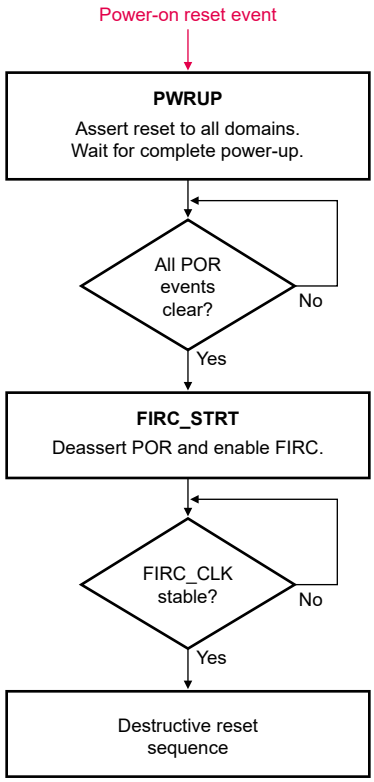


Figure 120. POR flow

30.4.2 Destructive reset

The chip enters a destructive reset sequence after the POR sequence or any destructive reset event completes. Destructive reset sequence illustrates the destructive reset sequence and Destructive sequence stage description discusses the stages of destructive sequence.

30.4.2.1 Destructive reset sequence

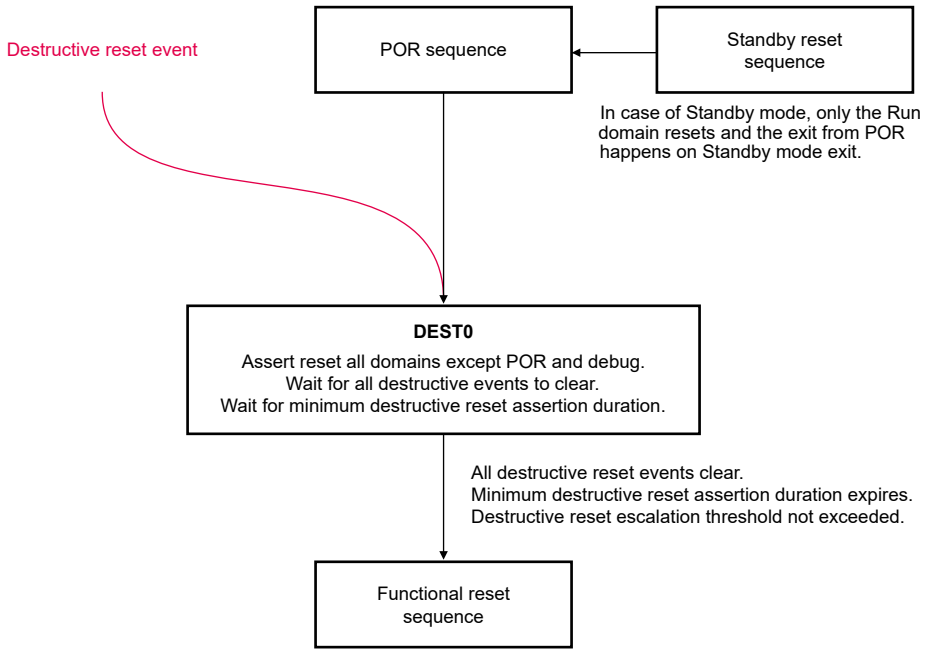


Figure 121. Destructive reset sequence

30.4.2.1.1 Destructive sequence stage description

Table 153. DEST0 description

Stage	Process
DEST0	1. Asserts reset to the entire chip, except logic running on POR. 2. Waits for all the destructive reset events to clear. 3. Waits for the minimum destructive reset assertion duration of eight FIRC_CLK cycles. 4. Deasserts after stage completion.

30.4.2.2 Destructive reset event bypass

This chip supports a destructive reset event demotion mechanism that the application software configures. The destructive reset bypasses and an interrupt event occurs (demotion). Table 154 discusses details related to GPR configuration and corresponding interrupt identification.

A successful chip operation is not guaranteed if a destructive reset event is bypassed.

Table 154. Destructive reset event bypass

Destructive reset event	Destructive reset event description	DCM field to bypass reset event	NVIC interrupt ID
MC_RGM.DES[PLL_LOL]	PLL loss of lock	DCM.DCMRWP3[9]	212

30.4.3 Functional reset

The chip enters the functional reset sequence when any of the following events occur:

- Functional reset
- POR or destructive reset (after the DEST0 stage completion)

On any functional reset event, the chip starts a functional reset entry sequence before the functional reset asserts and ensures the stability of logic running on a destructive reset and POR. On a destructive reset event or POR events the functional reset entry sequence does not execute.

The functional reset exit sequence consists of steps that ensure proper initialization of the chip after functional reset recovery.

30.4.3.1 Functional reset sequence

Functional reset flow illustrates the functional reset flow and Functional reset sequence descriptions discusses the functional reset stages and their descriptions.

Stages FUNC0 to FUNC6 present the functional reset entry sequence. It occurs on any functional reset event before the functional reset. In other words, when a functional reset event occurs, MC_RGM holds the asserted reset and executes the functional reset entry sequence. After the sequence completes, MC_RGM resets the chip, which remains in Run mode during the functional reset entry sequence.

Stages FUNC7 to FUNC11 present the functional reset exit sequence, which occurs after a functional reset event before deasserting the chip reset. This includes handshaking with the flash memory and analog blocks, ensuring correct operation after reset exit.

30.4.3.1.1 Functional reset flow

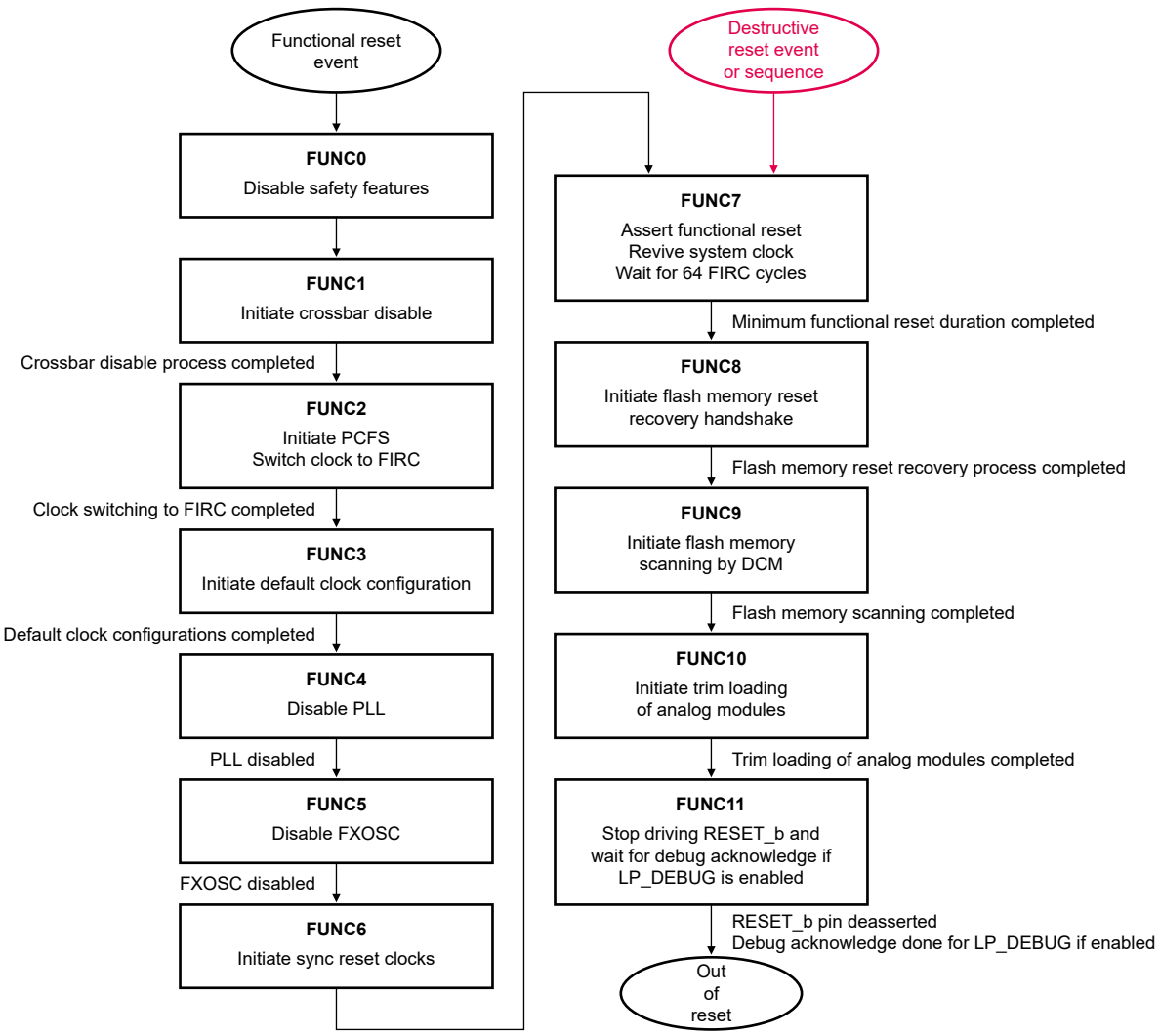


Figure 122. Functional reset flow

30.4.3.2 FUNC9 and FUNC10 stage bypass for faster Standby mode exit

The chip supports optional bypassing of the FUNC9 and FUNC10 (only FIRC and PMC phases) stages on Standby mode exit to considerably reduce the Standby mode exit duration. This feature is recommended only for Standby mode exit and must be configured on:

- Standby mode entry sequence
- SW3
- Disabled on Standby mode exit

See the "Faster Standby recovery" section in the "Power Management" chapter for more information.

30.4.3.3 Standby reset sequence

The logic within Run domain is reset additionally apart from the reset sequence in the chip standby entry sequence, wherein all the resets assert (POR, destructive, and functional) to the Run domain logic. The chip standby entry sequence does not have any impact on the logic in Standby power domain.

Wake-up from Standby mode removes the resets to the Run domain in the standby entry sequence.

See the "Peripheral reset status" section in the "Reset Overview" chapter for the logic present in Run domain.

30.4.3.4 Reset function redirection

Resets may escalate or demote to an IRQ, depending on the chip configuration.

30.4.3.4.1 Functional reset demotion to an interrupt

This chip supports the reset sequence demotion feature for functional resets. You can configure a functional reset to create an interrupt instead of a reset (see the MC_RGM chapter for details).

30.4.3.4.2 Reset escalation

The chip supports the reset escalation feature. If multiple functional or destructive resets occur, the related reset can escalate to a higher priority reset sequence (see the "Functional reset escalation" and "Destructive reset escalation" sections in the MC_RGM chapter).

30.4.3.4.2.1 Destructive reset escalation

You can enable destructive reset escalation by configuring a DCF client. You must also configure the destructive count threshold in MC_RGM.DRET (see "Destructive Reset Escalation Enable Register (DEST_RST_ESC):dcf_client_dest_rst_esc" in the DCF file attached to this document). The escalation event can individually be enabled or disabled for each reset source, and the fields in the dcf_client_dest_rst_esc register correspond with the fields in MC_RGM.DES register.

After being configured, MC_RGM immediately asserts a destructive reset escalation when the destructive event count reaches the threshold count in MC_RGM.DRET[DRET]. When the destructive and escalation reset assert, the reset sequence immediately enters the DEST0 state. The reset sequencing remains in DEST0 until a POR event occurs. If enabled, the destructive reset escalation counter increments with each destructive reset event. The application software clears the destructive reset escalation counter by writing any value to MC_RGM.DRET[DRET].

NOTE

You can configure GPR settings to allow demotion of destructive resets to interrupts instead of escalation (DCMRWP3[DEST_RST9_AS_IPI]). See Destructive reset event bypass for more information.

30.5 Reset timing diagram

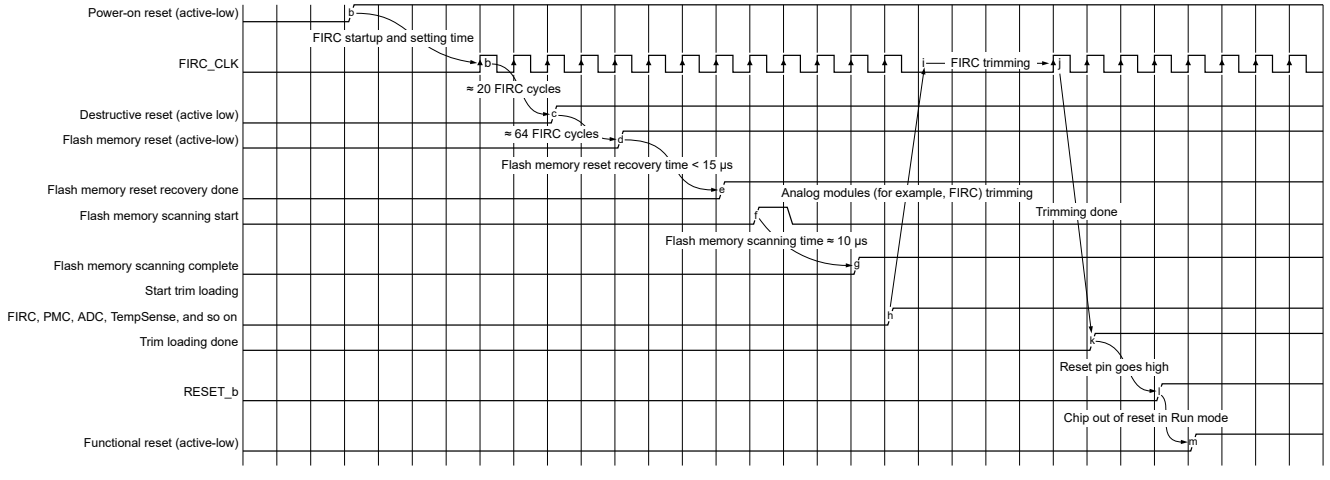


Figure 123. Reset timing diagram

30.6 Chip status after reset deassertion

Table 155. Chip status after reset deassertion

Function or feature	After POR deassertion	After destructive reset deassertion	After functional reset deassertion
Clock sources	• FIRC_CLK and SIRC_CLK on • Others off	• FIRC_CLK and SIRC_CLK on • Others off	• FIRC_CLK and SIRC_CLK on • SXOSC_CLK same as before functional reset • FXOSC_CLK and PLL_PHIn_CLK off
Clock selection	FIRC_CLK	FIRC_CLK	FIRC_CLK
Clock dividers	Default configuration	Default configuration	Default configuration
Reset status flags	MC_RGM.DES[F_POR] equals 1, others equal 0	MC_RGM.DES[F_DR_n] equals 1, others equal 0	MC_RGM.FES[F_FR_n] equals 1, others equal 0
MC_ME previous mode	Reset	Reset	Reset
FCCU fault information	Cleared	Cleared	Retained
HSE_B RAM	Initialized	Initialized	Initialized
SRAM content	Invalid	Invalid	Retained
DCF configurations in DCM	Reset value	Existing loaded value (reset value after POR)	Reloaded from flash memory
Cores	All off	All off	HSE_B core on, others off
Logic on POR[1]	Out of reset with default configuration	Out of reset with default configuration	Out of reset with default configuration
Logic on destructive reset [1]	Under reset with default configuration	Out of reset with default configuration	Out of reset with default configuration
Logic on functional reset [1]	Under reset with default configuration	Under reset with default configuration	Out of reset with default configuration

1. For the list of peripherals affected by POR, destructive reset, and functional reset events, see Module reset status.

30.7 Module reset status

Table 156. Module reset status

Module instances[1]	Destructive	Functional	Power domain[2]	Part of LBIST[3]
MC_RGM	Y	Y	Standby	No
PRAMC	Y	Y	Run	Yes
PFC	Y	Y	Run	Yes
SIUL_VIRTWRAPPER_PDAC0	Y	Y	Run	No

Table continues on the next page...

Table 156. Module reset status (continued)

Module instances[1]	Destructive	Functional	Power domain[2]	Part of LBIST[3]
SIUL_VIRTWRAPPER_PDAC1	Y	Y	Run	No
SIUL_VIRTWRAPPER_PDAC2	Y	Y	Run	No
SIUL_VIRTWRAPPER_PDAC3	Y	Y	Run	No
DCM	Y	Y	Run and Standby[4]	No
TRGMUX	Y	Y	Run	No
WKPU	Y	Y	Standby	No
CMU_F*x*_[0:5]	Y	Y	Run	CMU 0-3: No CMU 1-2: Yes CMU 4-5: Yes
FIRC	Y[5]	Y[5]	Standby	No
FXOSC	Y	Y	Standby	No
MC_CGM [6]	Y	N	Run	No
MC_ME	Y	Y	Run	No
PLL	Y	Y	Run	No
Configuration GPR	Y	Y	Run	No
eMIOS 0-2	Y	Y	Run	No
PIT_0	Y	Y	Standby[7]	No
PIT_[1:2]	Y	Y	Run	No
PIT_[3]	Y	Y	Run	No
FlexCAN_[0:5]	Y	Y	Run	No
FlexIO	Y	Y	Run	No
LPUART_[0:15]	Y	Y	Run	No
LPI2C_[0:1]	Y	Y	Run	No
LPSPI_[0:5]	Y	Y	Run	No
QuadSPI	Y	Y	Run	No
SAI_[0:1]	Y	Y	Run	No
ADC_[0:2]	Y	Y[10]	Run	No
LPCMP_[0:2]	Y	Y	Standby	No
TempSense	Y	Y[10]	Run	No
CRC	Y	Y	Run	Yes
FCCU (+FOSU)	Y	N	Run	Yes
STCU2	Y	N	Run	No
HSE_B MUA-MUB	Y	Y	Run	No

Table continues on the next page...

Table 156. Module reset status (continued)

Module instances[1]	Destructive	Functional	Power domain[2]	Part of LBIST[3]
MU_3 MUA-MUB	Y	Y	Run	No
MU_4 MUA-MUB	Y	Y	Run	No
JDC	Y[8]	Y[8]	Run	No
DMAMUX_[0:1]	Y	Y	Run	No
PMC	Y[9]	N	Standby	No
Flash memory	Y	Y[10]	Run	No
SIRC	Y[5]	Y[5]	Standby	No
SXOSC	Y[11]	N	Standby	No
BCTU	Y	Y	Run	No
LCU[0:1]	Y	Y	Run	No
RTC	Y	N[12]	Standby	No
EMAC	Y	Y	Run	No
HSE_B	Y	Y	Run	No
SWT_0	Y	Y	Standby	No
SWT_1	Y	Y	Run	Yes
STM_0	Y	Y	Run	No
STM_1	Y	Y	Run	No
MSCM	Y	Y	Run	No
ERM_0	Y	Y	Run	Yes
EIM_0	Y	Y	Run	Yes
eDMA	Y	Y	Run	Yes
JTAGC	N	N	Run	No
MDM_AP	Y	N	Run	No
APB_AP	Y	N	Run	No
Cortex-M7_0	Y	Y[13]	Run	No
Cortex-M7_1	Y	Y[13]	Run	No
Cortex-M7_0 AHB-AP	Y	N	Run	No
Cortex-M7_0 AHB-AP	Y	N	Run	No
MC_PCU	Y	N	Standby	No

Legends:	
Y	The entire module resets on this particular reset.
Y	Only a portion of the module resets on this particular reset.

Table continues on the next page...

Table 156. Module reset status (continued)

Module instances[1]	Destructive	Functional	Power domain[2]	Part of LBIST[3]
N	No portion of the module resets on this particular reset			

1. All the modules listed in this table are reset on a POR event. See the memory map file attached to this document for the availability of modules across various parts in the MWCT2xxxS family.
2. The modules in the RUN domain get reset on standby exit. The modules in STANDBY domain are not impacted by standby exit and retain their contents. However in case of standby exit via functional reset or destructive reset event, the corresponding flops within the STANDBY domain modules will also get reset.
3. The modules in the LBIST logic get reset on selftest completion.
4. Flash memory scanning logic is available in the Run domain. GPRs and LC decode logic are available in Standby domain.
5. All memory-mapped registers are on functional reset. The trimming logic is on destructive reset. Rest of counter and other stuff is on POR.
6. During functional reset stages FUNC2 and FUNC3 (see the Functional reset sequence descriptions for functional reset stage descriptions), MC_CGM.MUX_*n*_CSC and MC_CGM.MUX_*n*_DIV_*m* are automatically set to their default values. The default value of the MC_CGM.MUX_*n*_CSC[SELCTL] selects FIRC_CLK as the source clock for all multiplexers. The default value of the MC_CGM.MUX_*n*_DIV_*m* is register instance specific (see the "MC_CGM register descriptions" section in the Clock Generation Module (MC_CGM) chapter).
7. Only PIT_0 supports the RTI feature, and exists in the Standby domain.
8. The system domain is reset on functional reset. A POR will reset it completely.
9. PMC registers are reset on a destructive reset except PMC.LVSC, which is reset only by own PMC.LVSC[PORF] flag. The LVR and POR logic is reset on an LVR or own PORF (see PMC.CONFIG and PMC.LVSC descriptions for details).
10. Reset on a functional reset; however, reset recovery occurs before the chip functional reset recovery, at the functional reset exit sequence start, for proper trim scanning and loading.
11. SXOSC is reset on a destructive reset so that the RTC operates properly across a functional reset.
12. RTC operates during a functional reset.
13. The functional reset maps to nSYSRESET to Arm Cortex-M7 and the destructive reset maps to nPORESET to Arm Cortex-M7. See the Cortex-M7 TRM for further description on part of logic on different domains within Cortex-M7.

30.8 System RAM retention across functional reset

System RAM retains content during functional reset through the crossbar halt handshake. The system crossbar halts during the functional reset entry sequence. Therefore, the accesses do not cause any content corruption (see Functional reset sequence for more information).

Follow this sequence for the crossbar halt handshake (see Halt handshake using the daisy-chaining method for gasket locations):

1. Send a halt request to the HSE_B AXBS, DMA AXBS, and EMAC IAHB bridges in parallel.
2. Wait for a halt acknowledgement from HSE_AXBS and DMA AXBS.
3. Send a halt request to the DMA IAHB and HSE_B IAHB gaskets.
4. Wait for a halt acknowledgement from all the gaskets listed in the aforementioned steps.
5. Send a halt request to the system AXBS.
6. Wait for a halt acknowledgement from the system AXBS.
7. Send a halt request to a peripheral AXBS.
8. Wait for a halt acknowledgement from a peripheral AXBS.
9. Send a halt request to the TCM IAHB and QSPI IAHB gaskets.
10. Wait for a halt acknowledgement from all the gaskets listed in the aforementioned steps.
11. Send a halt request to the AIPS0 IAHB and AIPS1 IAHB gaskets.
12. Wait for a acknowledgement from all of the gaskets listed in the aforementioned steps.
13. Send a halt request to TCM AXBS.
14. Wait for a halt acknowledgement from TCM AXBS.

NOTE

AXBS halt handshake sequence is automatically performed by the hardware.

After this halt sequence completes, the crossbar halt acknowledgement sequence also completes and the chip proceeds to the FUNC1 stage in the functional reset entry sequence.

NOTE

RAM retention is supported across the functional reset event for system RAMs only and not for HSE_B or peripheral memories.

30.8.1 Halt handshake using the daisy-chaining method

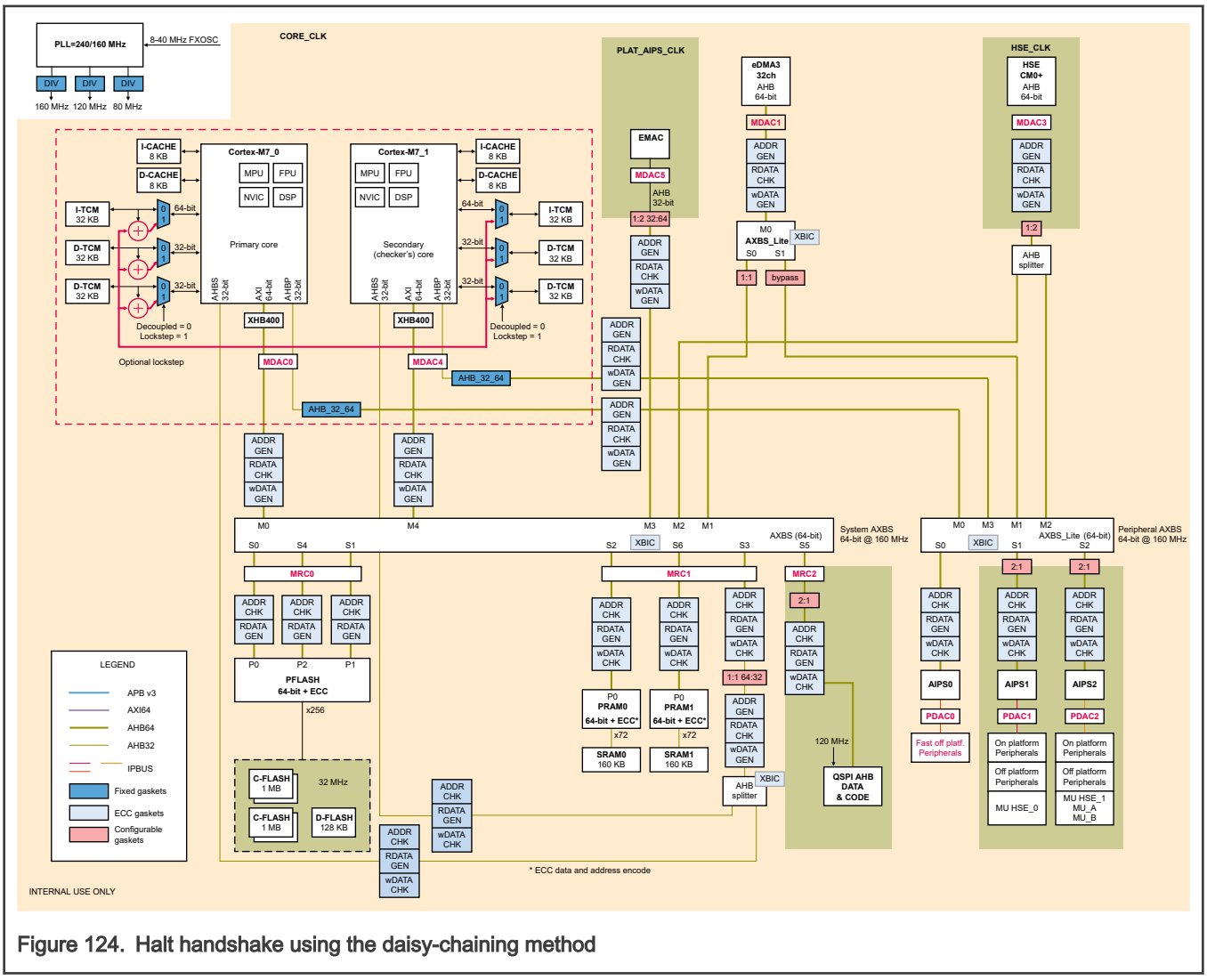


Figure 124. Halt handshake using the daisy-chaining method

30.9 Pad state during reset and after reset

SIUL2 controls the GPIO functionality. It sets to its default state on a functional reset, and ensures that every pad initializes to its default state (see the default configurations and reset states of the chip's GPIO in the IOMUX file attached to this document).

30.10 Reset pin

This chip contains a bidirectional reset pin (RESET_b) indicating the reset state. RESET_b multiplexes with the other functions on port PTA5 (see the IOMUX file attached to this document).

The DCF configuration controls the multiplexing capabilities of RESET_b. The default configuration of the RESET_b port is that of a dedicated reset signal (for example, not multiplexed. See the DCF clients file attached to this document for more information).

The RESET_b pin offers the following uses if you configure it for the reset functionality:

- Acts as an external destructive reset source
- Acts as an indicator for the chip reset sequence for both functional and destructive reset sequences

NOTE

MC_RGM.FES[F_EXR] captures an externally sourced RESET_b assertion for a destructive reset.

The RESET_b pin also indicates an internally asserted reset to external modules. It has a weak internal pullup. In normal Run mode, it keeps the chip out of reset.

30.10.1 Reset pin control during self-test

You can write 1 to MC_RGM.ERCTRL[ERASSERT] to assert RESET_b (writes only in Supervisor mode), before LBIST or MBIST executes. MC_RGM then asserts RESET_b and tristates the GPIO pins placing them in a safe state. Tristating GPIO ensures a safe state for the chip pins when LBIST or MBIST executes. Following BIST, the chip executes a reset sequence. The chip configures again for the application software, before executing the safety function. MC_RGM.ERCTRL[ERASSERT] clears on a functional reset. See Figure 8 for an illustration.

NOTE

Writing 1 to SIUL2.MSCR*n*[SMC] enables the GPIO pins and the chip continues its normal I/O functionality.

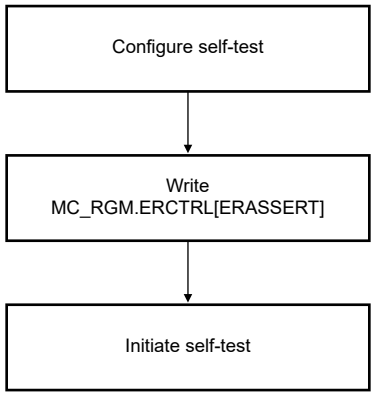


Figure 125. Reset pin control before self-test

Multiple chip configuration scenarios cause RESET_b to react differently after self-test completion:

- You can write 1 to MC_RGM.ERCTRL[ERASSERT] causing RESET_b to assert. This assertion does not impact the reset sequence, and the reset indicates that the chip is not available in Functional mode (although the chip is not in reset sequence).

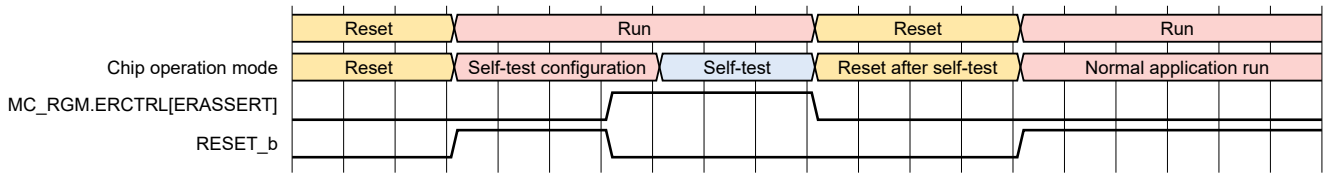


Figure 126. Reaction when MC_RGM.ERCTRL[ERASSERT] = 1

- If MC_RGM.ERCTRL[ERASSERT] = 0 (the default value), the RESET_b pin goes low after self-test completes. After self-test, the chip undergoes a functional reset in which the chip hardware writes 0 to MC_RGM.ERCTRL[ERASSERT]. The RESET_b pin goes high after the reset deasserts.

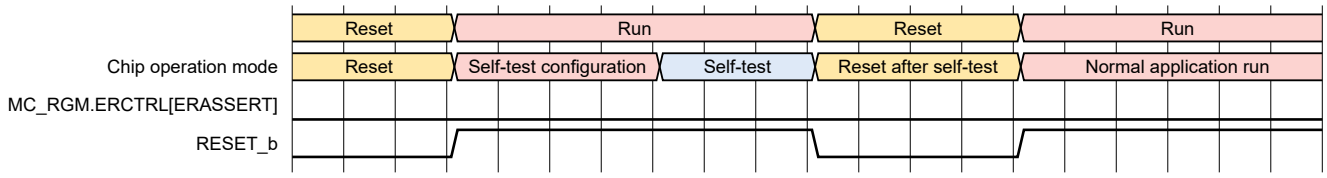


Figure 127. Reaction when MC_RGM.ERCTRL[ERASSERT] = 0

- If MC_RGM.ERCTRL[ERASSERT] = 0 (the default value), but you write 1 to MC_RGM.FBRE[ST_DONE], the RESET_b pin does not assert after self-test completes.

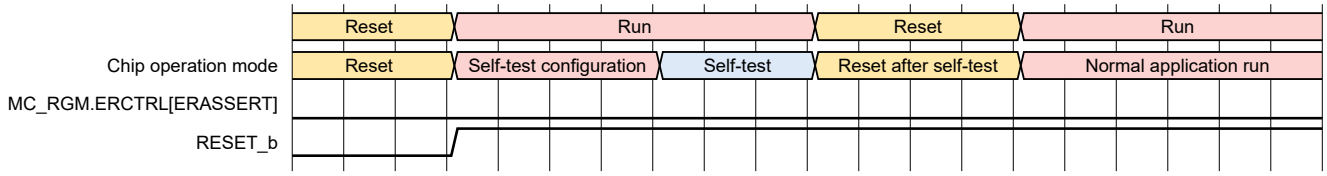


Figure 128. Reaction when MC_RGM.ERCTRL[ERASSERT] = 0 and MC_RGM.FBRE[ST_DONE] = 1

30.11 Reset control in Lockstep mode

In Lockstep mode, a two-cycle delayed lockstep implementation controls the reset to both the application cores, Cortex-M7_0 and Cortex-M7_1. This lockstep implementation means Cortex-M7_0 starts two clock cycles before Cortex-M7_1. You can configure the dcf_client_utest_misc[LOCSTEP_EN] field to control the lockstep (see the DCF clients file attached to this document for more information).

The Cortex-M7 cores consist of two reset domains:

- PORESETn (see PORESETn control in Lockstep mode)
- SYSRESETn (see SYSRESETn control in lockstep)

.

30.11.1 PORESETn control in Lockstep mode

PORESETn includes debug modules that work across chip warm resets. The PORESETn domain resets on any destructive reset event. PORESETn deasserts after two cycles of reset deassertion synchronization delay, as soon as destructive reset sequence exits. In Lockstep mode, the reset deassertion to Cortex-M7_1 is further delayed by two CORE_CLK cycles as shown in PORESETn control in Lockstep mode.

30.11.1.1 PORESETn control in Lockstep mode

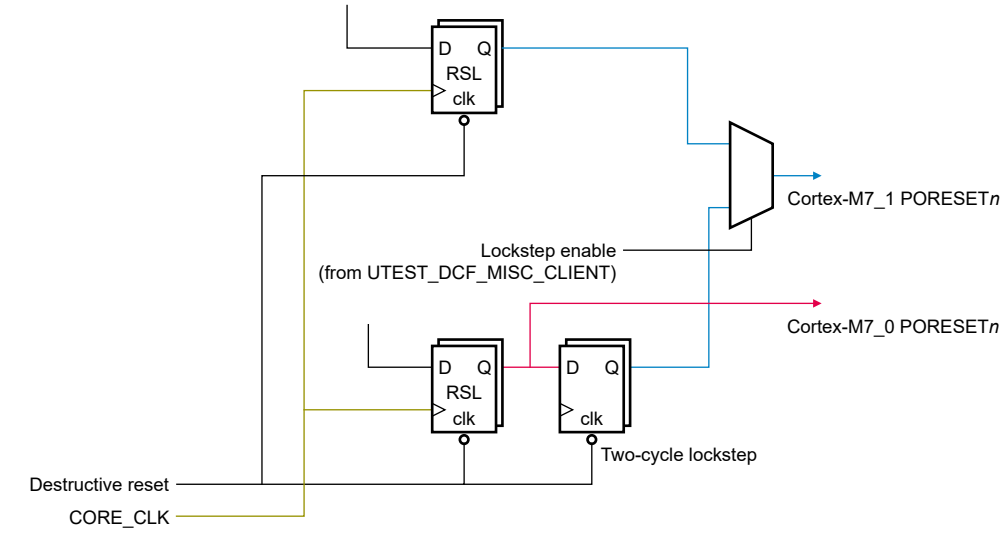


Figure 129. PORESETn control in Lockstep mode

30.11.2 SYSRESETn control in lockstep

Most of the components within the Cortex-M7 cores reside in the SYSRESETn domain, except the debug modules. These components reset on any functional reset event. The SYSRESETn reset remains gated even if it exits the functional reset until MC_ME.PRTN0_COREn_PCONF[CCE] enables the core clocks. The debugger can hold the core's SYSRESETn domain in reset, as described in section Application core debug from first instruction.

After a chip's functional reset deasserts, the core clocks are functional, and there is no gating from the debugger, the application core's SYSRESETn is released after a two-cycle reset deassertion synchronizer delay. In Lockstep mode, the reset deassertion to Cortex-M7_1 is delayed by two CORE_CLK cycles, as shown in SYSRESETn control in Lockstep mode.

30.11.2.1 SYSRESETn control in Lockstep mode

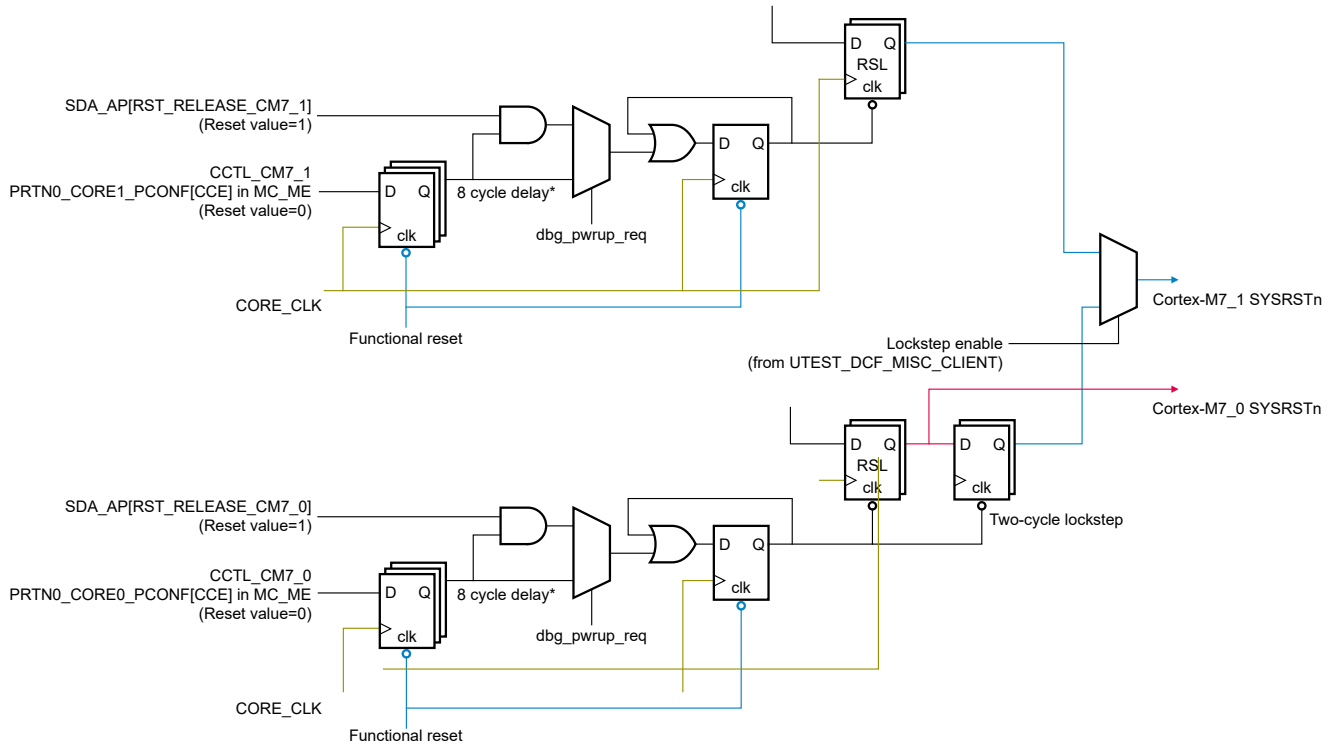


Figure 130. SYSRESETn control in Lockstep mode

30.12 Application core debug from first instruction

The chip supports application core debug starting on the first instruction. This is supported on any functional reset and through a RESET_b pin destructive reset for debug from destructive reset onwards using the following sequence (see SYSRESETn control in lockstep for more information on implementation).

30.12.1 Debug from first instruction on power-up or destructive reset onwards

1. When RESET_b is held low, the chip goes through a destructive reset sequence.
2. When you pull down the RESET_b pin, the chip goes through a destructive reset sequence, exits from the destructive reset, and remains in functional reset, until the reset pin is released.
3. When the RESET_b pin remains pulled down, the debugger configures SDA_AP.SDAAPRSTCTRL[RSTRELTLCM70], SDA_AP.SDAAPRSTCTRL[RSTRELTLCM71] and SDA_AP.SDAAPRSTCTRL[RSTRELTLCM72] according to the requirements. To enable debug from first instruction, you write 0 to these fields. To disable, you write 1. By default, the debug disables from the first instruction.
4. On releasing the reset pin, the chip exits from the functional reset sequence and HSE_B starts executing.
5. The PORESETn deassertion occurs for the application cores simultaneously (in case of lockstep, Cortex-M7_1 PORESETn deasserts after a two-cycle delay). The corresponding core clock enable gates the deassertion of SYSRESETn to cores. The SYSRESETn to the cores deasserts as follows.
6. On the basis of whether the debugger authentication is performed via challenge, response, or neither of them, you can consider these scenarios:
    a. Debugger not connected (DBG_PWRUP_REQ = 0):
        i. BAF or FW writes to the clock enable control field in MC_ME for application cores, that is, to MC_ME.PRTN0_CORE0_PCONF[CCE] for Cortex-M7_0, MC_ME.PRTN0_CORE1_PCONF[CCE] for

Cortex-M7_1 and MC_ME.PRTN0_CORE2_PCONF[CCE] for Cortex-M7_2 or to all the three fields as required.

ii. The clock enables in the subsequent clock cycle.

iii. SYSRESETn deasserts after ten clock cycles (adding eight clock cycles of delay ensures that reset always deasserts when the clock is available; two cycle of reset deassertion delay). In case of lockstep, Cortex-M7_1 reset is delayed by two additional clock cycles.

b. Debugger connected (DBG_PWRUP_REQ = 1):

i. BAF or FW writes to the clock enable control field in MC_ME for application cores, that is, to MC_ME.PRTN0_CORE0_PCONF[CCE] for Cortex-M7_0,MC_ME.PRTN0_CORE1_PCONF[CCE] for Cortex-M7_1 and MC_ME.PRTN0_CORE2_PCONF[CCE] for Cortex-M7_2 or to all the three fields as required.

ii. The clock is enabled.

iii. The debugger waits for sometime (at least ten clock cycles) and then writes 1 to SDA_AP.SDAAPRSTCTRL[RSTRELTLCM70] for Cortex-M7_0, SDA_AP.SDAAPRSTCTRL[RSTRELTLCM71] for Cortex-M7_1 and SDA_AP.SDAAPRSTCTRL[RSTRELTLCM72] for Cortex-M7_2, for releasing SYSRESETn of the corresponding core.

iv. SYSRESETn deasserts after ten clock cycles (adding eight clock cycles of delay ensures that the reset always deasserts when the clock is available; two cycle of RSL delay). In case of lockstep, the Cortex-M7_1 reset is delayed by two additional clock cycles.

NOTE

- CortexM7_1 is present in MWCT2D16S.

30.12.2 Debug from first instruction on functional reset onwards

If you enable debug, on functional reset, the debug connection is retained. That is because the complete debug infrastructure is on destructive reset domain.

30.12.3 Debug on standby exit by wake-up or functional reset

If you enable debug before standby entry, then the Standby domain stores the final status. The configuration for reset release is reset (SDA_AP.SDAAPRSTCTRL[RSTRELTLCM70], SDA_AP.SDAAPRSTCTRL[RSTRELTLCM71] and SDA_AP.SDAAPRSTCTRL[RSTRELTLCM72]. The reset value of the three fields is 1), which defaults to debug from the first instruction disabled.

You must configure the reset release on standby exit. It uses MC_RGM's low-power debug protocol before ungating the reset deassertion.

30.13 Glossary

DCF	Device configuration format
POR	Power-on reset

Chapter 31
Boot Overview

31.1 Overview

31.1.1 Introduction

This chapter describes the system boot sequence and provides details about boot options.

After the hardware reset sequence completes, the only CPU available is in the HSE subsystem that is referred to as the HSE CPU.

The HSE CPU starts executing firmware code in the HSE code flash memory from a fixed location that contains the SBAF code. This code provides the boot sequence until the control is passed, based on the type of boot:

- Nonsecure boot: Passes control to the customer software that executes on one or all the application cores.
- Secure boot: Passes control to the HSE firmware running on the HSE CPU.

31.2 Appendix

SBAF takes into account the following scenarios to prevent stuck in reset of chip:

- After observing the eight functional resets in the chip, BAF enters Recovery mode sequence to recover the application core’s failing status.
- SBAF does not allow the chip's LC to advance to the OEM_PROD or IN_FIELD stage, if the application does not program CUST_DB_PSWD/A.
- SBAF boots the application from the system-RAM during recovery mode sequence to avoid unpredictable behavior.

31.2.1 Features

The features of SBAF are as follows:

- Supports secure and nonsecure boot modes
- Supports application boot core selection
- Allows chip LC advancement
- Supports debug authorization
- Supports XRDC configuration

31.3 Chip configuration

This section describes the chip configuration details for MWCT2xxxS variants after you program SBAF and clear the HSE firmware feature flag.

If the security is enabled, see the HSE Firmware Reference manual for more information. Please contact NXP sales executive for details.

31.3.1 Memory map

This section explains the memory sections used by SBAF in various configurations.

31.3.1.1 Configuration details when the HSE firmware usage feature flag is disabled

Table 157. Configuration details when the HSE firmware usage feature flag is disabled

Memory section	MWCT2015S	MWCT2016S, MWCT2D16S	MWCT2D17S
Flash memory	1 MB	2 MB	4 MB
IVT locations in priority order	0040_0000h, 0048_0000h, 1000_0000h	0040_0000h, 0050_0000h, 1000_0000h	0040_0000h, 0050_0000h, 0060_0000h, 0070_0000h, 1000_0000h
Reserved	004F_4000h to 004F_FFFFh (48 KB)	005F_4000h to 005F_FFFFh (48 KB)	007F_4000h to 007F_FFFFh (48 KB)
Application flash memory area	0040_0000h to 004F_3FFFh (976 KB)	0040_0000h to 005F_3FFFh (2000 KB)	0040_0000h to 007F_3FFFh (4048 KB)
Application data flash memory	1000_0000h to 1000_FFFFh (64 KB)	1000_0000h to 1001_FFFFh (128 KB)	

31.3.1.2 Configuration details when the HSE_B firmware usage feature flag is enabled

The following table explains the memory sections used by the SBAF in case HSE firmware feature flag is enabled in UTEST.

Table 158. Configuration details when the HSE_B firmware usage feature flag is enabled

Memory section	MWCT2015S	MWCT2016S, MWCT2D16S,	MWCT2D17S
Flash memory	1 MB	2 MB	4 MB
IVT locations in priority order	0040_0000h, 0048_0000h, 1000_0000h (256 Bytes)	0040_0000h, 0050_0000h, 1000_0000h (256 Bytes)	0040_0000h, 0050_0000h, 0060_0000h, 0070_0000h 1000_0000h (256 Bytes)
Reserved	004D_4000h to 004F_FFFFh (176 KB)	005D_4000h to 005F_FFFFh (176 KB)	007D_4000h to 007F_FFFFh (176 KB)
Application flash memory area when full memory "HSE firmware" is present	0040_0000h to 004D_3FFFh (848 KB)	0040_0000h to 005D_3FFFh (1824 KB)	0040_0000h to 007D_3FFFh (3920 KB)
Application data flash memory	1000_0000h to 1000_FFFFh (64 KB)	1000_0000h to 1001_5FFFh (88 KB)	

31.3.1.3 AB swap configuration

Table 159. AB swap configuration

Memory section	MWCT2015S	MWCT2016S, MWCT2D16S	MWCT2D17S
Flash memory	1 MB	2 MB	4 MB
IVT locations in priority order	0040_0000h, 1000_0000h	0040_0000h, 1000_0000h	0040_0000h, 0050_0000h, 1000_0000h
Reserved code area in active block	0045_4000h to 0047_FFFFh (176 KB)	004D_4000h to 004F_FFFFh (176 KB)	005D_4000h to 005F_FFFFh (176 KB)
Reserved code area in passive block	004D_4000h to 004F_FFFFh (176 KB)	005D_4000h to 005F_FFFFh (176 KB)	007D_4000h to 007F_FFFFh (176 KB)
Application flash memory area in active block	0040_0000h to 0045_3FFFh (336 KB)	0040_0000h to 004D_3FFFh (848 KB)	0040_0000h to 005D_3FFFh (1872 KB)
Application flash memory area in passive block	0048_0000h to 004D_3FFFh (336 KB)	0050_0000h to 005D_3FFFh (848 KB)	0060_0000h to 007D_3FFFh (1872 KB)
Application data flash memory	1000_0000h to 1000_FFFFh (64 KB)	1000_0000h to 1001_FFFFh (128 KB)	

31.4 Common configuration pertaining to the chip

31.4.1 Feature configuration in CUST_DEL Device Life Cycle

The following table describes the status of various features when chip's LC is in the CUST_DEL stage. The application software configures features directly or indirectly.

Table 160. Feature configuration in CUST_DEL Device Life Cycle

Feature	Configuration information	Status	Configurability
OTA functionality		Disabled	Yes Application requests the HSE firmware or SBAF to enable this feature.
HSE firmware usage feature flag	Indicates whether the firmware installation is allowed in the chip. By default, this flag is unprogrammed, and SBAF assumes that the firmware installation is not allowed.	Unprogrammed	Yes To enable this feature, program in the UTEST location (see UTEST flag description for more information).

Table continues on the next page...

Table 160. Feature configuration in CUST_DEL Device Life Cycle (continued)

Feature	Configuration information	Status	Configurability
SBAF firmware		Programmed	No
HSE firmware	You must program the HSE firmware. An encrypted and signed firmware image is always delivered to you.	Not programmed	Yes SBAF can install this feature when the application software requests.
Image vector table		Not programmed	Yes The application software can program this feature.
SWT0		Disabled	By SWT bit in boot configuration word.
Boot sequence	Boot sequence is a nonsecure boot, which means, the SBAF boots the application without any authentication.	Nonsecure boot	Yes It can be changed to secure boot by programming the BOOT_SEQ bit in IVT.
Life Cycle		Customer delivery	Yes Advance by SBAF or HSE firmware when requested by application software.
Application core enablement status	Applications cores are booted in recovery mode sequence at address 2040_0100h.	Recovery mode sequence	Yes Program the required fields in the IVT to enable single or all application cores at the required address.
FIRC frequency value		48 MHz	Yes The application can configure after SBAF moves to WFI.
Application debug authorization mode	This mode is password-based. You can program the configuration in UTEST to change the mode to Challenge Response.	Password-based approach	When the HSE firmware feature flag clears, you cannot change Debug Authorization mode. When the HSE firmware feature flag is set, you can request the HSE firmware to change

Table continues on the next page...

Table 160. Feature configuration in CUST_DEL Device Life Cycle (continued)

Feature	Configuration information	Status	Configurability
			Debug Authorization mode to Challenge Response mode.
Application core debug status	Debug of application cores is enabled in the customer delivery life cycle.	Enabled	No

31.4.2 UTEST memory location usage by SBAF

Table 161. UTEST memory location usage by SBAF

Start address	End address	Size (bytes)	Description	Programmed by	Write protected
1B00_0000h	1B00_0007h	8	HSE firmware feature usage flag. See UTEST flag description for more information.	Application software	No
1B00_0040h	1B00_0047h	8	Unique Chip Identifier (UID)	NXP	No
1B00_0050h	1B00_0057h	8	FXOSC configuration flag See UTEST flag description for more information.	Application software	No
1B00_0080h	1B00_009Fh	32	Debug password (CUST_DB_PSWD_A) After the HSE firmware usage feature flag clears, SBAF uses this location to run the debug authorization feature. SBAF copies this value in the application expected response register, which derives the HSE expected response register. The size of this register is 16 bytes, and 1B00_0090h – 1B00_009Fh is reserved. After the HSE firmware usage feature flag sets, the HSE firmware programs the password at a different location. DCM scans the password during reset only and retains the password in standby.	Application software	No

See the DCF clients file attached to this document for more information.

31.4.3 UTEST flag description

31.4.3.1 HSE firmware usage feature flag

This flag indicates to SBAF that the application intends to use the HSE firmware on the chip. By default, this flag is unprogrammed, and SBAF assumes that the HSE firmware installation is not allowed in the secure samples. However, if application allows the installation of the HSE firmware, the HSE firmware usage feature flag can be programmed in the UTEST at the 1B00_0000h location.

Table 162. HSE firmware usage feature flag

Field type	Description	Remarks
Size	64 bits	
Default value	0xFFFFFFFFFFFFFFFF	SBAF does not allow the installation of HSE firmware.
UTEST location	0x1B000000	
Configurability	Application software in CUST_DEL lifecycle	
Enable flag	Program any value other than the default value	This enables the installation of the HSE firmware.

31.4.3.2 Crystal oscillator configuration flag

See the below table for description of the flag:

Table 163. Flag fields

Field type	Description
Size	64 bits
Default value	FFFF_FFFF_FFFF_FFFFh (Boot via FIRC)
UTEST location	1B00_0050h
Configurability	Application software in any LC

Table 164. Crystal oscillator configuration flag in UTEST

	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	FXOSC_ENABLE_MAGIC_NUMBER															
W																
	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R																
W																

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GMSEL				EOCV											
W																
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CRYSTAL_OSCILLATOR_FREQUENCY															
W																

Table 165. Crystal oscillator configuration bit definition

Field	Description
63 - 33	FXOSC_ENABLE_MAGIC_NUMBER AAAA_5555h – Enable (FXOSC.CTRL[OSCON] = 1h) FFFF_FFFFh – Disable (FXOSC.CTRL[OSCON] = 0h)
31 - 28	Crystal overdrive protection (GMSEL) For values = 0h or Fh Uses the default value, Ch.
27 - 20	End Of Count Value (EOCV) For value = 0h Uses the default value, 9Dh.
19 - 16	Reserved
15 - 0	CRYSTAL_OSCILLATOR_FREQUENCY Frequency value of used external crystal oscillator in kHz. The valid crystal frequency range is 8000–40000 kHz.

31.5 Image vector table

The following section describes the fields in the image vector table that the application programs. SBAF scans the IVT after the chip is out of reset. The structure is 256 bytes in size. This structure contains application cores start addresses. IVT must be programmed at least at one of the locations described in Chip configuration.

After reset, SBAF searches for the first valid IVT starting from the lowest address. If there are multiple valid IVT at IVT locations at the same time, the IVT with lowest address is used.

Table 166. Image vector table

Address offset	Size in bytes	Content	Comments
00h	4	Image vector table marker	marks the starting of the image vector table location. Its value must be 5AA5_5AA5h.

Table continues on the next page...

Table 166. Image vector table (continued)

Address offset	Size in bytes	Content	Comments
04h	4	Boot configuration word	Indicates the configuration word that allows you to select the various configurations in which you can boot the chip. See the upcoming section for more information.
08h	4	Reserved	
0Ch	4	Cortex-M7_0 core start address	Specifies the boot address of the Cortex-M7_0 core in the code flash memory area. It must honor core Vector Table Offset Register (VTOR) alignment restrictions. SBAF uses this field only when the HSE firmware usage feature flag is enabled and BOOT_SEQ field is 0.
10h	4	Reserved	
14h	4	Cortex-M7_1 core start address	Specifies the boot address of the Cortex-M7_1 core in the code flash memory area. It must honor core VTOR register alignment restrictions. SBAF uses this field only when the HSE firmware usage feature flag is enabled and BOOT_SEQ field is 0.
18h	4	Reserved	
1Ch	4	Cortex-M7_2 core start address	Specifies the boot address of the Cortex-M7_2 core in the code flash memory area. It must honor core VTOR register alignment restrictions. SBAF uses this field only when the HSE firmware usage feature flag is enabled and BOOT_SEQ field is 0.
20h	4	Reserved	
24h	4	Address of LC configuration	Specifies the address of the configuration word that allows you to advance the LC. See the upcoming section for more information.
28h	216	Reserved	

31.5.1 Boot configuration word

This register informs SBAF to allow booting of selected applications.

Table 167. Boot configuration register

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Table continues on the next page...

Table 167. Boot configuration register (continued)

R											APP_SWT_INIT	RESERVED	BOOT_SEQ	CM7_2_ENABLE	CM7_1_ENABLE	CM7_0_ENABLE
W																

Table 168. Boot configuration register field definition

Field	Description
31 – 6	Reserved
5	APP_SWT_INIT Controls the SWT0 enablement before passing the control to the application core(s). 0b - Disables 1b - Enables SBAF initializes SWT0 before enabling the application cores. SBAF scans this field only when the HSE firmware usage feature flag is enabled and the BOOT_SEQ field is 0.
4	Reserved
3	BOOT_SEQ Controls the boot flow of the application when HSE Firmware usage feature flag is enabled. 0b - Nonsecure boot: SBAF starts the application image without any authentication in parallel to the HSE firmware. 1b - Secure boot: The HSE firmware executes the application image after authentication. SBAF only starts the HSE firmware after successful authentication.
2	CM7_2_ENABLE Indicates whether the Cortex-M7_2 application core clock is gated after boot. 0b - Gated 1b - Ungated
1	CM7_1_ENABLE Indicates whether the Cortex-M7_1 application core clock is gated after boot. 0b - Gated 1b - Ungated
0	CM7_0_ENABLE Indicates whether the Cortex-M7_0 application core clock is gated after boot. 0b - Gated 1b - Ungated

31.5.2 Address LC configuration word

This field allows you to advance the LC. To advance LC, you must program a valid 32-bit wide value at an address given in IVT at an address offset of 24h. This address must be 4 bytes aligned and should not lie in HSE reserved area.

The following table shows the valid values for advancement in the next LCs. For all other values at the given address, LC advancement is not allowed.

Table 169. Valid values for LC advancement

Life cyle stage	Valid values for LC advancement
OEM_PROD	DADA_DADAh
IN_FIELD	BABA_BABAh

Depending on the HSE firmware feature flag, the application password on the location must program before LC advancement; otherwise, SBAF does not attempt LC advancement.

The chip provides an LC mechanism for an irreversible progression of restrictions to access the chip's security-related content. You cannot reverse the chip's LC, so it is only possible to mature the chip. SBAF advances the chip through the LC:

- CUST_DEL --> OEM_PROD or IN_FIELD
- OEM_PROD --> IN_FIELD

To advance the LC through SBAF involves inserting the LC configuration word address in the IVT. SBAF issues a destructive reset on successful advancement. If the chip is found in the same LC as the IVT indicated, you can ignore LC advancement.

31.5.3 Structure definition of image vector table

Application can use the following structure to configure the IVT.

```
typedef const struct image_vector_table
{
uint32_t Header; /*Header of IVT Structure */
uint32_t BootConfig; /*Boot Configuration Word */
const uint32_t Reserved1; /* Reserved */
const uint32_t * CM7_0_StartAddress; /*Start Address of Application on CM7_0 Core */
const uint32_t Reserved2; /* Reserved */
const uint32_t * CM7_1_StartAddress; /*Start Address of Application on CM7_1 Core */
const uint32_t Reserved3; /* Reserved */
const uint32_t * CM7_2_StartAddress; /*Start Address of Application on CM7_2 Core */
const uint32_t * Reserved4; /* Reserved */
const uint32_t * LCConfig; /*Address of LC configuration Word */
uint8_t Reserved5[216]; /* Reserved */
}ivt_t;
```

31.6 Boot flow

Below diagram explains boot sequence flow of SBAF.

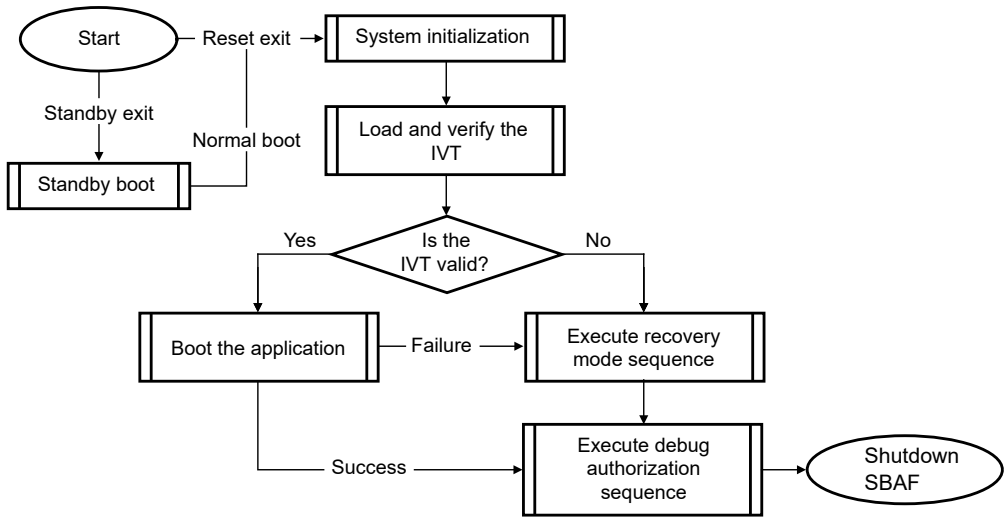


Figure 131. Boot flow

31.7 Standby boot

SBAF supports boot from standby exit. Chip register DCM.DCMRWF5 (address 402AC610h) supports boot on standby exit. This register clears on POR. See the "Device Configuration Module (DCM)" chapter in the MWCT2xxxS reference manual for more information.

You must program this register before entering Standby mode.

There are two types of boot mode on exit from standby.

- Fast Standby
- Normal boot on exit from standby

In Fast Standby mode, the SBAF boots the Cortex-M7_0 core and halts the HSE CPU. The flow of Standby Boot is explained below:

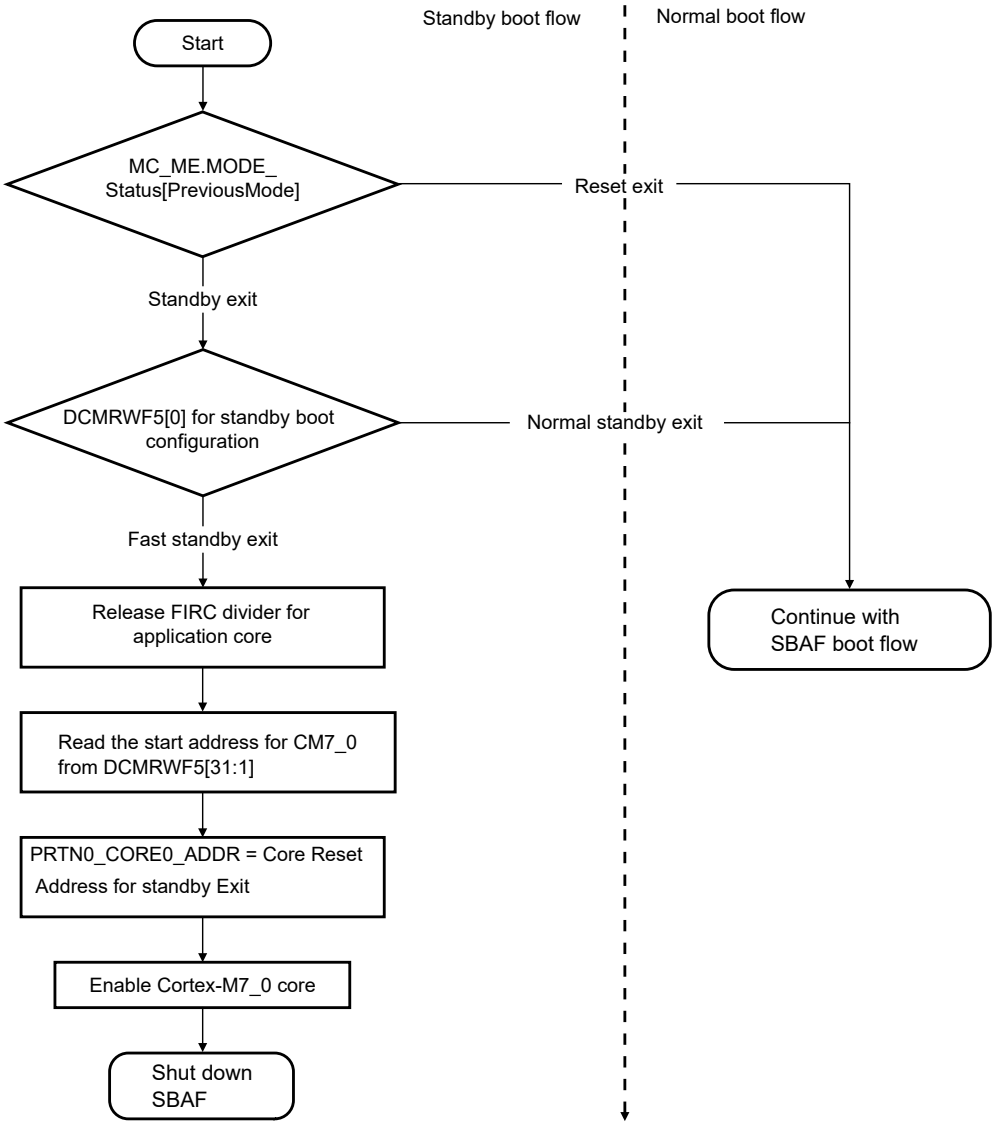


Figure 132. Standby boot flow

31.8 Reduced clock mode configuration

If you use clocking option B (Reduced clock mode configuration), the application sets the “dcf_client_utest_misc” DCF record to enable Reduced Clock mode. See the DCF clients file attached to the MWCT2xxxS reference manual for more information on DCF records.

After reset, SBAF checks for DCMROF21[HSE_CLK_MODE_OPTION]. If you configure this field for clocking option B, the SBAF configures the following dividers in MC_CGM:

- MUX_0_DC_1
- MUX_0_DC_2

Below figure explains steps for reduced clock mode configuration steps by SBAF.

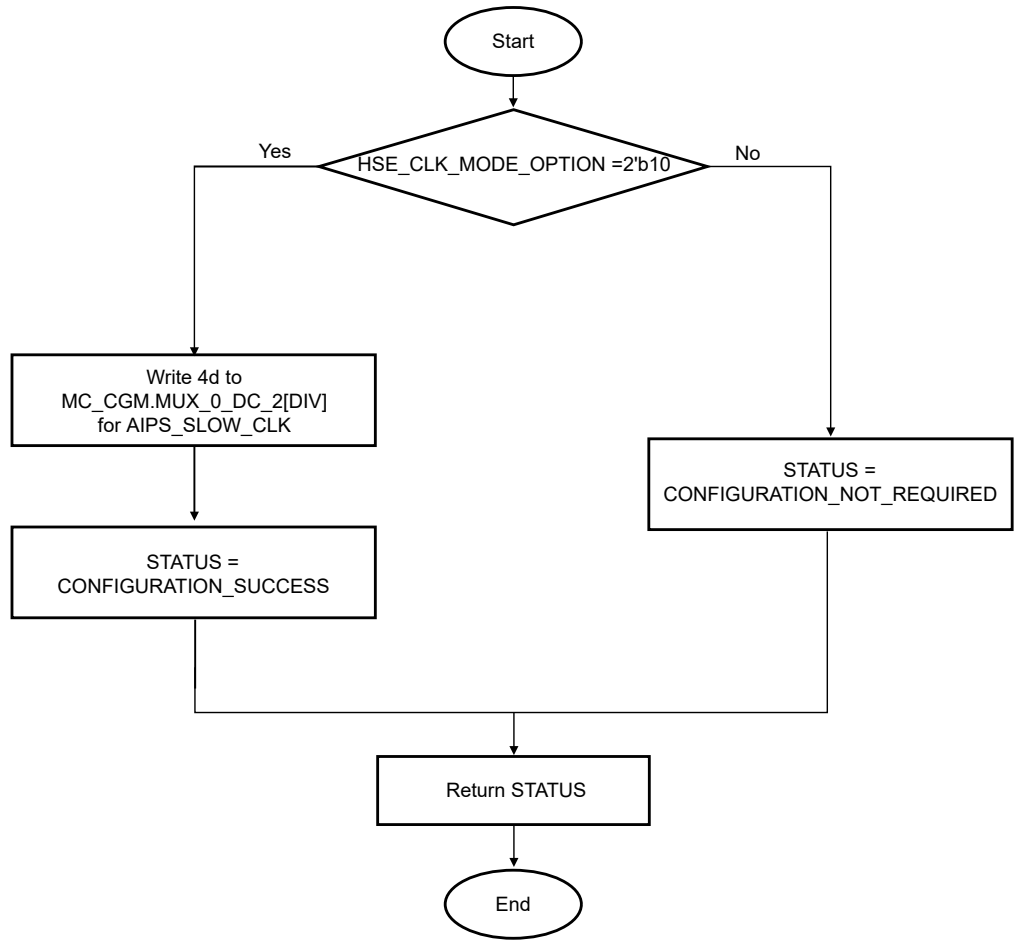


Figure 133. Reduced clock mode configuration

31.9 Debug authorization

You must program CUST_DB_PSWD_A at location 1B00_0080h. The application core debug is always password-based if the HSE firmware usage feature flag is cleared. See UTEST memory location usage by SBAF for more information.

31.10 FIRC divider register control

HSE.CONFIG_REG_GPR, at register address 4039C064h, controls the FIRC divider. This register is write-protected by default for the FIRC divider, and you cannot modify its settings.

After SBAF executes WFI, it provides write access to HSE.CONFIG_REG_GPR[FIRC_DIV_SEL], and you can configure this register. Before accessing this register, you must wait for the SBAF to enter WFI by reading core status register of HSE CPU (PRTN0_CORE2_STAT).

31.11 Application boot

The HSE firmware is responsible for securely booting the application image and not the SBAF. If secure boot is not requested (by writing 0 to the BOOT_SEQ field in Boot configuration word in IVT structure), SBAF always loads the application image, without authentication, in all chips, LC. By default, SBAF releases the reset of every core that you configure to enable.

Following flow chart explains the steps of application boot performed by SBAF.

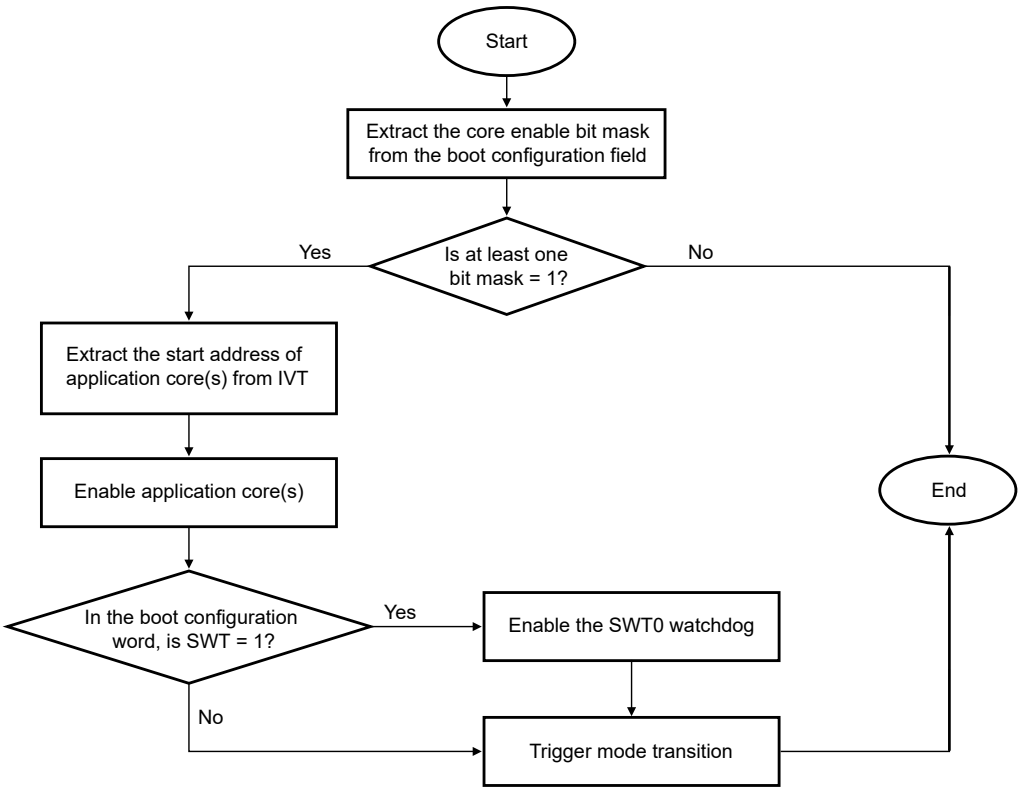


Figure 134. Application boot sequence

Before configuring HSE_CLK, you must wait for the SBAF to enter WFI by reading core status register of HSE CPU (PRTN0_CORE2_STAT).

31.12 Recovery mode sequence

This feature allows you to program the application image in LC = MCU_PROD and LC = CUST_DEL and debug the reason of corruption of IVT and re-program the IVT in other LC.

The following scenarios take place when SBAF executes the recovery mode sequence.

- Valid IVT is not found (corrupted or not programmed).
- SBAF does not boot the application.
- Boot configuration word in IVT does not program the application enablement field.
- The application issued more than eight functional resets and Disable Recovery mode on functional reset field is not set in DCM.DCMRWP1.
- The application issued more than eight destructive resets and Disable Recovery mode on destructive reset field is not set in DCM.DCMRWP1.

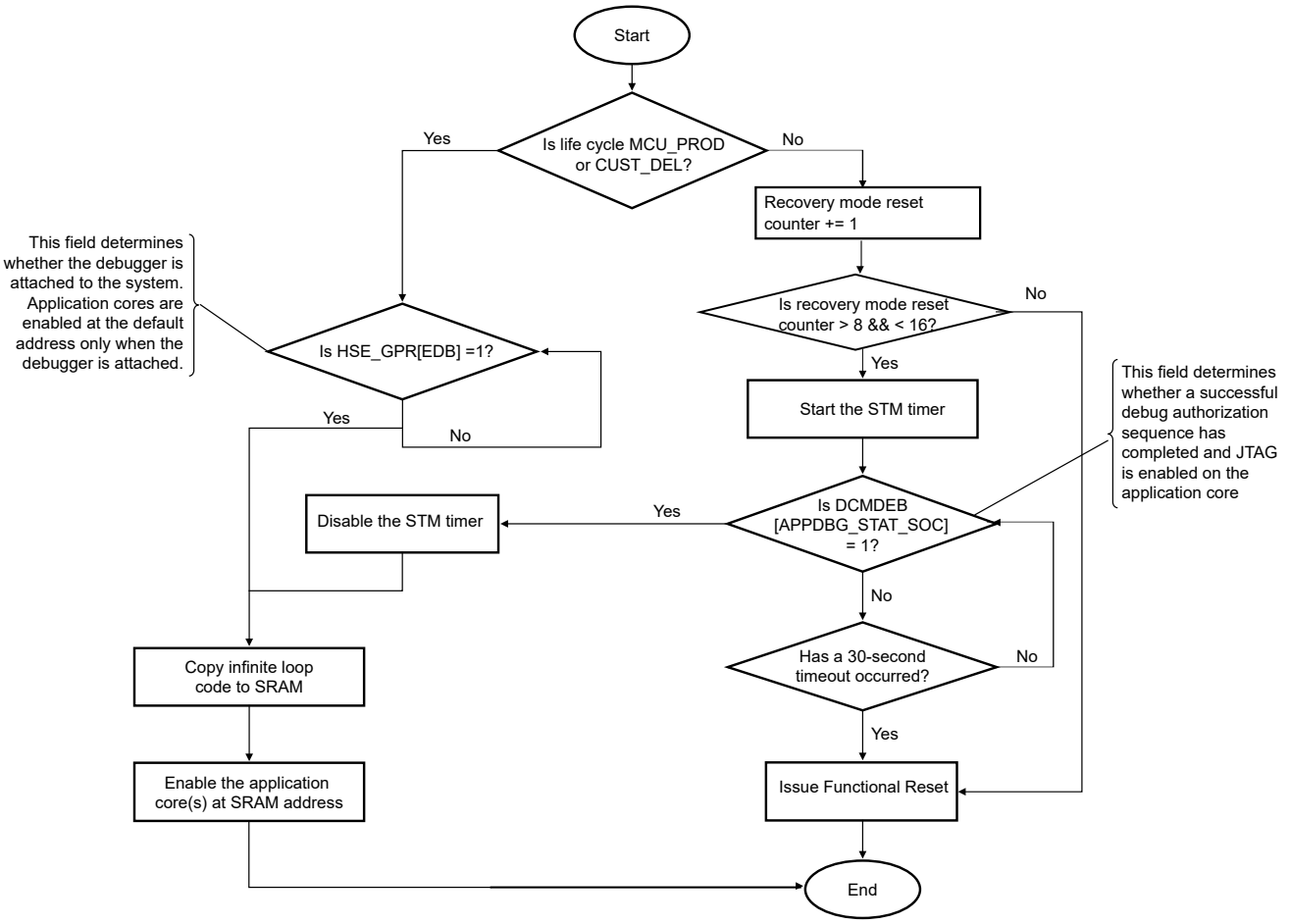


Figure 135. Recovery mode sequence

The infinite loop with the WFI code copies to SRAM1 that is 2040_0100h, and the code size is 16 bytes. To prevent prefetching errors in the application cores, place the infinite loop instruction at 2040_012Ch.

31.13 XRDC configuration

SBAF ensures that it has access to its resources and the application does not have access to area reserved for SBAF.
The following sections describe the XRDC default configuration values when you clear the HSE firmware usage feature flag.
Application should configure its XRDC and enable the XRDC by itself as SBAF does not enable the XRDC.

31.13.1 XRDC configuration of MDAC

Following table list down the default configuration of MDAC by SBAF.

Table 170. XRDC configuration of MDAC

Serial No.	Chip variant	MDAC register HSE CPU	Value	Domain number assigned to HSE CPU
1	MWCT201xS	MDA_W0_3_DFMT0	C000_0001h	1
2	MWCT2D1xS	MDA_W0_3_DFMT0	C000_0002h	2

31.13.2 XRDC configuration of MRC

Following table list down the MRC used by SBAF.

Table 171. XRDC configuration of MRC

MRC number	Region descriptor number	Remarks
0	14	Reserved for application
0	15	Reserved for application

31.13.3 XRDC configuration of PDAC

SBAF configures and lock the following peripherals for its use. SBAF provides all permissions to all domains.

Table 172. XRDC configuration of PDAC

Serial No.	Peripheral name	Peripheral PDAC number
1	Flash controller alternate	155
2	Flash memory alternate	188
3	HSE_GPR	231

31.14 BAF flash programming controls

The platform flash controller generates an exception of read-while-write if you simultaneously perform read and write on the same block. HSE CPU uses the Configuration PE Lock Register (CONFIG_PE_LOCK) in the HSE space to control the block program and erase for application.

BAF locks the high addressing code flash memory block during its execution, and it is cleared when the HSE CPU enters WFI. See the "Chip Configuration" chapter for the address of the high addressing code flash memory area.

Before programming, erasing, or executing from this address space, the application core polls for PRTN0_CORE2_STAT[WFI] to ensure that the HSE CPU is in the WFI state.

For boot sequence 1 or flash synchronization with the HSE firmware, see the HSE Firmware Reference Manual.

Below table explains PE lock bits setting in CONFIG_PE_LOCK register of HSE GPR during SBAF execution.

Table 173. PE lock fields setting in HSE.CONFIG_PE_LOCK

Chip	UTEST block	Data flash block	Code flash block 3	Code flash block 2	Code flash block 1	Code flash block 0
MWCT2015S	1	0	NA	NA	NA	1
MWCT2x16S	1	0	NA	NA	1	1
MWCT2D17S	1	0	1	0	0	1

31.15 Status registers for application usage

This section explains various status registers to provide status information to the application.

31.15.1 SBAF version information

The SBAF version is a 64-bit field. The application can read the SBAF from address 4039_C020h. The following table describes the version information.

Table 174. SBAF version information

Bits	Field name	Description
0 – 7	Reserved	Reserved
8 - 15	SOC_TYPE_ID	This field represents the SBAF firmware, which is targeted for MWCT2xxxS chip variant. Values of this field are: 0xB - used for HSE-B MWCT2014S 0xC - used for HSE-B MWCT2015S 0xD - used for HSE-B MWCT2x16S
16 - 31	FW_TYPE	This field represents the SBAF firmware type. Values of this field are: 0 – used for standard generic firmware targeting all customers 1-7 – Reserved >=8 used for Custom 1, Custom 2 (for example: Custom 1 = customer X's project A, Custom 2 = customer Y's project B)
32 - 39	Reserved	Reserved
40 - 47	BASELINE_NUMBER	Incremented when the compatibility with the previous version is broken.
48 - 55	INCREMENTAL_ NUMBER	Incremented when new features are added but compatibility kept.
56 - 63	RC_NUMBER	Release Candidate Number

31.15.2 DCM.DCMRWP1

The application can disable Recovery mode entry by SBAF after programming bits 23 and 22 of DCM.DCMRWP1.

Table 175. DCM.DCMRWP1 (address 0x402AC400)

Bits	Number of bits	R/W access by application	Description
24-31	8		Reserved
23	1	R/W	Disable Recovery Mode On Destructive Reset Indicates that this field resets by default, and SBAF allows Recovery mode sequence if the application issues > 8 destructive resets. The application can set this field to disable Recovery mode when the application issues > 8 destructive resets.
22	1	R/W	Disable Recovery Mode On Functional Reset

Table continues on the next page...

Table 175. DCM.DCMRWP1 (address 0x402AC400) (continued)

Bits	Number of bits	R/W access by application	Description
			Indicates that this field resets by default, and SBAF allows Recovery mode sequence if the application issues > 8 functional resets. The application can set this field to disable Recovery mode when the application issues > 8 functional resets.
21	1	R	Reserved
16-20	5	R	Recovery Mode Reset Counter Indicates that to enable Recovery mode functionality for the OEM_PROD and IN_FIELD LC stages, SBAF increments this counter when a functional or destructive reset is issued.
15	1	R	Reserved
11-14	4	R	Destructive Reset Counter Indicates that SBAF increments this counter when a destructive reset is issued.
0-10	11	R	Reserved

31.15.3 Status bits on HSE.GPR

SBAF writes HSE.GPR at address 4039_C028h to show status information as described in the following table.

Table 176. Status bits on HSE.GPR

Bit	Description
0	Indicates that SBAF presents and boots the HSE FW.
1-4	Reserved.
5	Indicates that SBAF boots the application cores in Recovery mode sequence.
6	Indicates that SBAF performs the debug authentication.
7-31	Reserved.

31.16 Interrupt and exception handling

- **Interrupt handling:** No special interrupt handling routines are required during the boot process. Interrupts are disabled during SBAF execution.
- **Exception handling:** SBAF enters the recovery mode sequence after enabling debug authorization. After eight consecutive functional resets or destructive resets from Application Firmware, the device enters the recovery mode sequence.
- **Boot target watchdog:** SBAF enables/disables SWT0 watchdog with default timeout, that is 25 ms according to the boot configuration word, before enabling the application core(s). It is expected that the application services this watchdog before expiration.

31.17 Hardware modules used by SBAF

- **MC_ME:** SBAF uses MC_ME to enable the application cores, mode switch, and other operations during its execution.

- **FXOSC:** SBAF configures FXOSC according to the crystal oscillator configuration flag in UTEST.
- **Clock Generation Module (MC_CGM):** SBAF configures MC_CGM in Reduced clock mode configuration.
- **DCM:** SBAF uses the DCM to identify the Life Cycle, standby boot mode configuration, lockstep, and clock mode configuration.
- **HSE_GPR:** SBAF configures hardware protection, program erase lock, FIRC divider control, and SBAF version number.
- **XRDC:** SBAF configures the XRDC module.
- **Flash Module:** SBAF always perform the write and erase operation on alternate interface. See Chip configuration for details of flash memory usage by SBAF.

31.18 Hardware IP registers details modified by SBAF

Below table summarizes the registers which are modified by SBAF when HSE Firmware usage feature flag is disabled.

Table 177. Hardware IP registers modified by SBAF

IP	Register Name	Default value (hex)	Modified value (hex)
MC_ME	MC_ME.PRTN1_COFB1_STAT	1CFE_2FFCh	FXOSC and MC_CGM clocks are enabled by default. However, for 120 MHz clock requests, SBAF ensures if the MC_CGM clock is enabled.
	MC_ME.PRTN0_COREx_ADDR	0040_0000h	SBAF updates this address if you request the application core 0 boot during the normal boot sequence in IVT or Recovery mode (0x2040012C).
	MC_ME.PRTN0_COREx_PCONF	0000_0000h	The core clock is enabled when the recovery mode sequence is executed or when the clock is enabled in Boot configuration word.
	MC_ME.PRTN0_COREx_PUPD	0000_0000h	
	MC_ME.CTL_KEY	0000_5AF0h	SBAF updates this register if SBAF boots any one of the application core. 0000_5AF0h and then 0000_A50Fh.
FXOSC	FXOSC.CTRL	019D_00C0h	See Crystal oscillator configuration flag
MC_RGM	MC_RGM.DRET	0000_0000h	0000_000Fh
SWT_0	SWT_0.CR	FF00_010Ah	FF00_000Bh, when SWT_0 is enabled in the boot configuration word and FF00_000Ah when the SWT_0 is disabled in the boot configuration word.

Table continues on the next page...

Table 177. Hardware IP registers modified by SBAF (continued)

IP	Register Name	Default value (hex)	Modified value (hex)
	SWT_0.IR	0000_0000h	0000_0001h

31.19 Glossary

BAF Boot assist flash

CUST_DEL Chip's life cycle stage, customer delivery

IN_FIELD Chip's life cycle stage, in field

IVT Image vector table

LC Chip's life cycle—limits by design the configuration and debug/test possibilities of the chip for in-field usage

MCU_PROD Chip's life cycle stage, MCU production

OEM_PROD Chip's life cycle stage, OEM production

OTA Over the air

SBAF Secure boot assist flash

Chapter 32
Reset Generation Module (MC_RGM)

32.1 Chip-specific MC_RGM information

32.1.1 MC_RGM configuration

The "Reset sources—POR, Destructive, and Functional" section of the "Reset Overview" chapter provides information about MC_RGM's reset sources. The chapter also provides details about the chip's reset architecture.

NOTE

The ERCTRL register configuration takes several cycles to be effective. Any further access to MC_RGM must happen after at least nine AIPS_SLOW_CLK cycles of writing to ERCTRL.

Table 178. Register fields and applicability

Register	Bit field	Chips where applicable
Functional /External Reset Status Register (FES)	SWT1_RST	MWCT2D17S, MWCT2D16S
Functional Event Reset Disable Register (FERD)	D_SWT1_RST	MWCT2D17S, MWCT2D16S

32.1.2 Functional reset entry timer implementation

The default timeout value of the functional reset entry timer is 2048 clocks of MC_RGM clock (FIRC). If the RGM entry sequence hangs, then POR_WDG would trigger and the status of this functional reset entry sequence timeout is indicated at chip-level status register. The status of timeout in such case, is indicated at DCM.DCMROPP1[28]. There is no impact to the device behavior if the functional reset entry sequence gets completed within the POR_WDG timeout window.

32.1.3 MWCT2xxxS reset state machine

The reset sequence of the MWCT2xxxS products is depicted in the figure below:

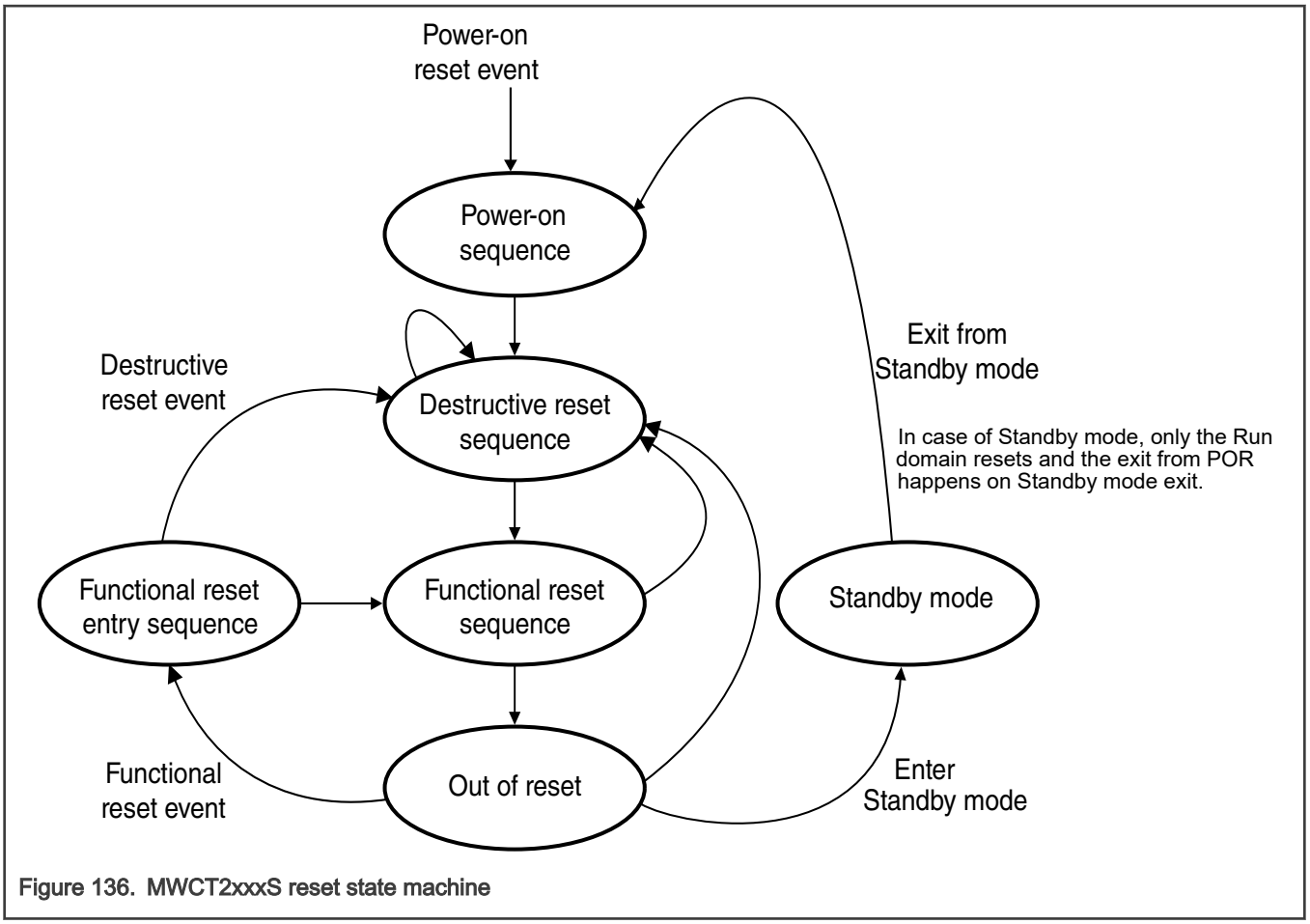


Figure 136. MWCT2xxxS reset state machine

32.2 Introduction

The Reset Generation Module (MC_RGM) centralizes the different reset sources and manages the reset sequence of the chip. It provides a register interface and the reset sequencer. There are various registers available in this module to monitor and control the chip reset sequence.

The following figure shows the block diagram of MC_RGM.

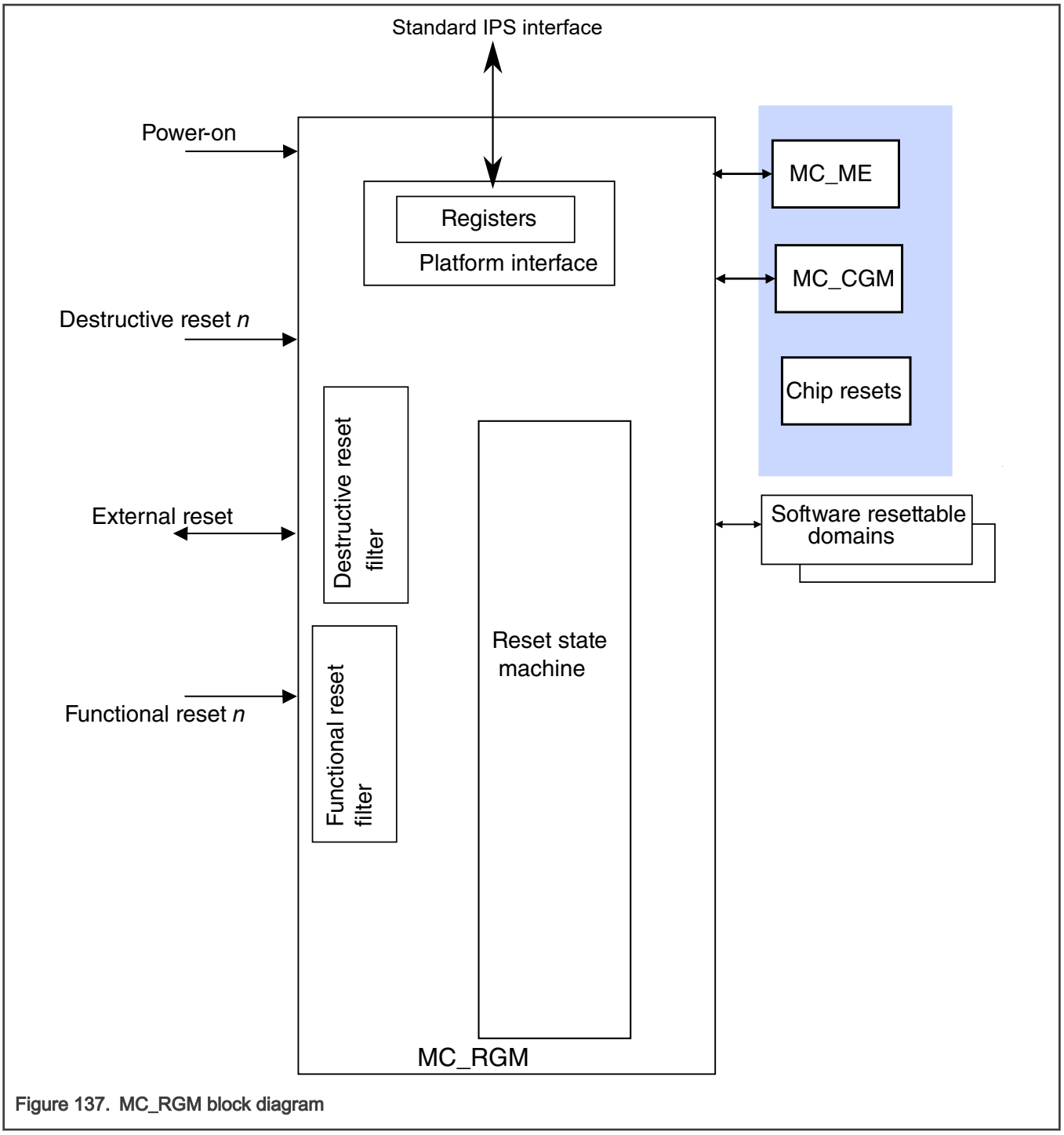


Figure 137. MC_RGM block diagram

32.3 Features

Here are the key features of MC_RGM:

- Destructive and functional reset management
- Capturing the reset sources for each reset sequence (reset status flags)
- Assertion the RESET_B pin to propagate the reset sequence out of chip
- Configurable escalation of recurring 'functional' resets to 'destructive' reset

- Configurable escalation of recurring 'destructive' resets to keep the chip in the reset state until the next power-on reset
- Software controllable reset assertion
- Pad safe state control generation

32.4 Reset sources

The reset sources are organized in three categories: power-on, destructive, and functional.

A power-on reset source is associated with an event typically related to power-up or low-voltage scenarios. When a power-on reset occurs, the full reset sequence is applied to the chip. This resets the full chip, including the MC_RGM, and the memory content must be considered to be invalid.

A destructive reset source is associated with an event related to a critical, usually hardware, error or dysfunction. When a destructive reset event occurs, the full reset sequence is applied to the chip. This resets the full chip ensuring a safe start-up state for both digital and analog modules, and the memory content must be considered to be invalid.

A functional reset source is associated with an event related to a less-critical, usually non-hardware, error or dysfunction. When a functional reset event occurs, a partial reset sequence is applied to the chip. In this case, most digital modules are reset normally, while the state of analog modules or specific digital modules as well as the system memory content is preserved.

32.5 External signal description

The MC_RGM interfaces with the RESET_B pin.

The following table describes the signals that are connected to the I/O pad ring.

Table 179. MC_RGM external signals

Signal name	Reset value	Description
RESET_B	0	Active low external reset. A bidirectional reset pin indicating the reset state.

32.6 RESET_B pin assertion and pin safe state control

The MC_RGM asserts the RESET_B pin when the device is in a reset sequence, and it remains asserted until the end of the reset sequence. During this reset sequence, most of the chip's pins are safe/pad stated according to the values shown in the IOMUX table/spreadsheet. Note that the safe state values may vary according to the reset sequence type.

In addition, the MC_RGM has a feature to assert the RESET_B pin through software, without initiating a reset sequence. This is achieved by writing 1b to the ERCTRL[ERASSERT]. When this occurs, most of the chip's pins are safe-stated according to the values shown in the IOMUX table/spread sheet. The RESET_B assertion and pin safe-stating remain active until the end of the next reset sequence.

This features has to be used only with selftest of the main reset domain.

32.7 Functional description

32.7.1 Reset state machine

The main role of MC_RGM is the generation of the reset sequence that ensures that the correct parts of the chip are reset based on the reset source event.

For each reset event, immediately after it is captured by the MC_RGM, the following takes place:

1. The corresponding reset event status bit is set in the MC_RGM_DES and MC_RGM_FES registers.
2. The pins are put into their default states

3. The RESET_B pin is asserted.

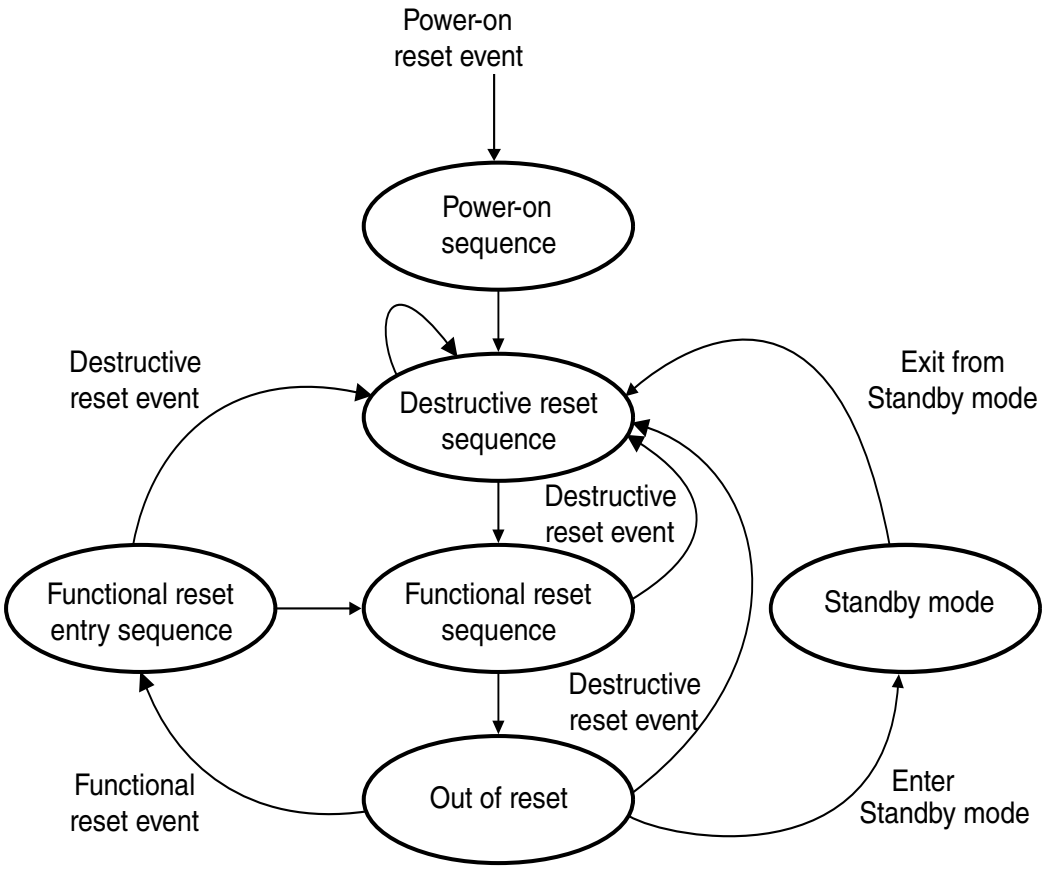


Figure 138. Reset sequence

NOTE

See chip-specific MC_RGM information for chip-specific reset sequence details.

32.7.1.1 Power-on reset sequence

A reset is always generated when the power-on reset source is asserted, and it has priority over all other reset sources. Such a power-on reset forces the reset state machine to enter the power-on sequence resulting in the assertion of all reset signals. The reset state machine starts progressing when the following two conditions are verified:

1. All the power-on reset events are cleared
2. The MC_RGM's clock source (the FIRC) has started up and stabilized

If a power-on reset event has occurred, the DES[F_POR] bit is set.

The power-on reset cannot be demoted by the software.

32.7.1.2 Destructive reset sequence

The 'Destructive reset sequence' is comprised of a number of phases, where DEST0 is the first phase and is followed by DEST1 and so forth.

This phase is entered immediately from any phase on a power-on, standby reset sequence, or enabled destructive reset event . A destructive reset counter starts immediately on entry in the DEST0 phase. The DEST0 state is exited to the DEST1 state on the rising edge of FIRC_CLK immediately after all of the following conditions have been established:

- The DEST0 duration time has expired
- All destructive reset inputs are cleared

The DEST0 state is immediately exited to the power-on state if a power-on reset event occurs.

The reset state machine exits the destructive reset sequence and enters the functional reset sequence when:

- All the destructive reset events are cleared.
- All the processes that take place during the destructive reset sequence have completed. For details, see Reset chapter.
- The 'destructive' reset escalator counter has not reached the value in DRET[DRET].

32.7.1.3 Functional reset sequence

There are two functional reset sequence, the functional reset entry sequence and the functional reset exit sequence.

32.7.1.3.1 Functional reset entry sequence

The functional reset entry sequence is only entered when a functional reset event occurs during the idle phase.

If a functional reset event occurs during an ongoing reset sequence, the corresponding event status flag is set, and the RESET_B pin is asserted per the reset event's configuration. However, the reset sequence is not influenced, and it continues to progress without interruption.

Functional reset is not asserted during functional reset entry sequence.

The functional reset entry sequence is exited to the DEST0 on the next rising edge of FIRC if a destructive reset event has occurred. The sequence immediately enters the power up sequence if a POR event occurs

In all other cases, the sequence exits to the first stage of functional reset exit sequence.

32.7.1.3.2 Functional reset exit sequence

The functional reset exit sequence is entered either on exit from the destructive reset sequence or on completion of the functional reset entry sequence. The reset state machine exits this sequence and enters the idle phase on verification of the following:

- All the functional reset events are cleared.
- All the processes that take place during the functional reset sequence have completed. For details, see Reset chapter.

If a functional reset event occurs during an ongoing reset sequence, the corresponding event status flag is set, and the RESET_B pin is asserted per the reset event's configuration. However, the reset sequence is not influenced, and it continues to progress without interruption.

32.7.1.4 Idle phase

This is the final phase and is entered on exit from the functional reset exit sequence . When this phase is reached, MC_RGM releases control of the system to the platform and waits for the new reset events that can trigger a reset sequence.

32.7.2 Destructive resets

A destructive reset indicates that an event has occurred after which critical register or memory content can no longer be guaranteed.

The status flag associated with a given destructive reset event (Destructive Event Status Register (DES)) is set when the destructive reset is asserted and the power-on reset is not asserted. It is possible for multiple status bits to be set simultaneously and the software determines which reset source is the most critical for the application.

The low-voltage detector threshold ensures that when the reset corresponding to the core supply low-voltage detect is enabled, the supply is sufficient to have the destructive event correctly propagated through the digital logic. Therefore, if a given destructive reset is enabled, MC_RGM ensures that the associated reset event is correctly triggered to the full system.

An enabled destructive reset triggers a reset sequence starting from the beginning of DEST0.

32.7.3 External reset

MC_RGM manages the external reset coming from RESET_B. The detection of a falling edge on RESET_B starts the reset sequence from the beginning of the destructive reset entry sequence.

The status flag associated with the external reset falling edge event (the FES[F_EXR]) is set when the external reset is asserted and the power-on reset is not asserted.

32.7.4 Functional resets

A functional reset indicates that an event has occurred after which it can be guaranteed that critical register and memory content is still intact.

The status flag associated with a given functional reset event (Functional /External Reset Status Register (FES)) is set when the functional reset is asserted and the power-on reset is not asserted. It is possible for multiple status bits to be set simultaneously and the software determines which reset source is the most critical for the application.

An enabled functional reset triggers a reset sequence starting from the beginning of the functional reset entry sequence.

32.7.5 Functional reset entry timer

The purpose of Functional Reset Entry Timer feature is to indicate any scenario wherein functional reset entry sequence does not get completed in prerequisite time .

When FRENTC[FRET_EN] is enabled, the functional reset entry timer starts from functional reset event point and gets terminated if the functional reset entry sequence is successfully completed or when it reached to the timeout value . It gets cleared on any destructive or POR reset events.

The timeout value is controlled by FRENTC[FRET_TIMEOUT]. The FRENTC[FRET_TIMEOUT] value should be greater than zero. If configured as zero, the default timeout value 200 gets used.

32.7.6 Alternate event generation

MC_RGM provides alternative events to be generated on reset source assertion. When a reset source is asserted, MC_RGM normally enters the reset sequence. Alternatively, it is possible for some reset source events to be converted from a reset to an interrupt request issued to the core. Alternate event selection for a given reset source is made through the RGM_FERD register as shown in the following table.

Table 180. Functional Reset Disable Register (RGM_FERD) field descriptions

RGM bit FERD value	Generated event
0	Reset
1	Interrupt request

The alternate event is cleared by deasserting the source of the request (that is, at the reset source that caused the alternate request) and also clearing the appropriate RGM_FES status bit.

32.7.7 RESET_B assertion control

The software indicates to the MC_RGM that the RESET_B is to be asserted by writing to the **ERASSERT** bit in the RGM_ERCTRL register. When this bit is set by the software, RESET_B gets asserted. Setting of this field does not impact the reset sequence in any way.

An example where the ERCTRL[ERASSERT] field could be set by the software is when entering the self test sequence, during which RESET_B is to be asserted. This indicates the chip is not available in the functional mode although a reset sequence is not in progress. The deassertion of RESET_B is not controlled by the software. Instead, the RESET_B pin remains asserted until the next time the chip exits a reset sequence.

ERASSERT bit is also cleared during the reset sequence.

MC_RGM asserts the external reset if the reset sequence is triggered by one of the following:

- A power-on reset
- A destructive reset event
- A functional reset event

In this case, external reset is asserted until all conditions for the exiting reset sequence have been met, with the exception of the RESET_B assertion check

NOTE

Software cannot request for external reset to be asserted in HK mode.

32.7.8 Functional reset escalation

Functional reset escalation can be used to generate a destructive reset if a number of functional resets is occurred between software writes to the RGM_FRET register. This function is enabled by writing a non-zero value to the FRET field of this register.

After the functional reset escalation is enabled, MC_RGM increases a counter on each functional reset that causes a reset sequence to be initiated (which means, entrance into FUNC0 from the IDLE phase). This counter is cleared on a write of any value to the RGM_FRET register and on any power-on or destructive reset. If the counter reaches the value in the FRET field of the RGM_FRET register, MC_RGM asserts a destructive reset.

The following figure shows the functional reset escalation counter.

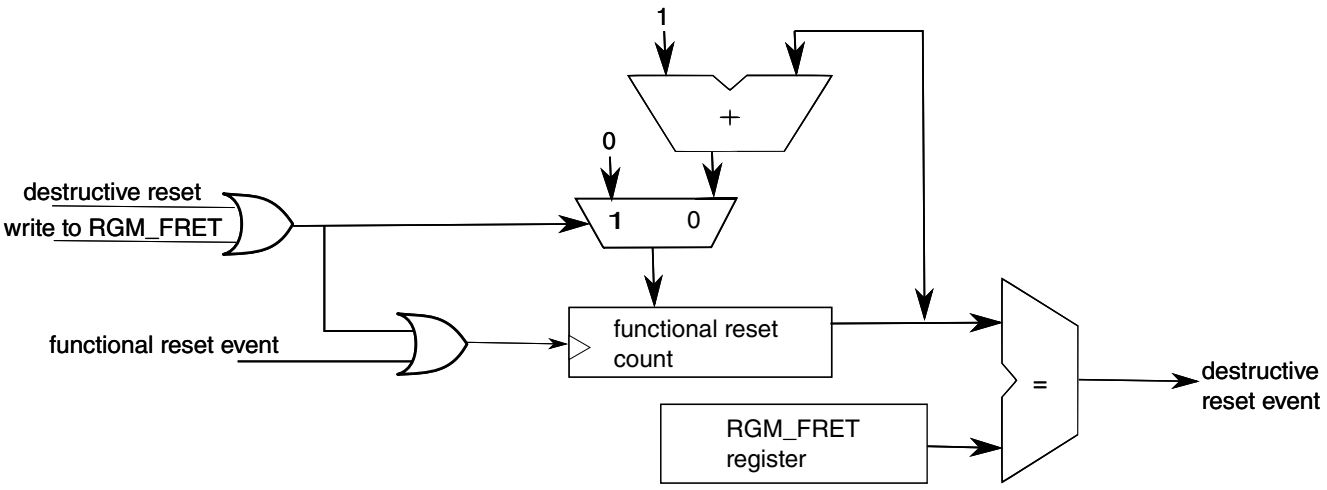


Figure 139. Functional reset escalation counter

NOTE

Functional counter increments for each reset source for which escalation is enabled. For details, see "Reset sources" table in the Reset chapter.

32.7.9 Destructive reset escalation

Destructive reset escalation can be used to keep the chip in the reset state until the power-on triggers a reset sequence if a number of destructive resets are occurred between software writes to the RGM_DRET register. This function is enabled by writing a non-zero value to the DRET field of this register.

After destructive reset escalation is enabled, MC_RGM increases a counter on each destructive reset that is enabled. . This causes a reset sequence to be initiated (that is, entrance into DEST0 from the idle phase or any other reset phase) or an ongoing reset sequence to restart (that is, entrance into DEST0 from any other reset phase). This counter is cleared on a write of any value

to the RGM_DRET register and on any power-on reset. If the counter reaches the value in the DRET field of the RGM_DRET register, MC_RGM enters reset DEST0 and stays there until the next power-on reset occurs .

The following figure shows the destructive reset escalation counter.

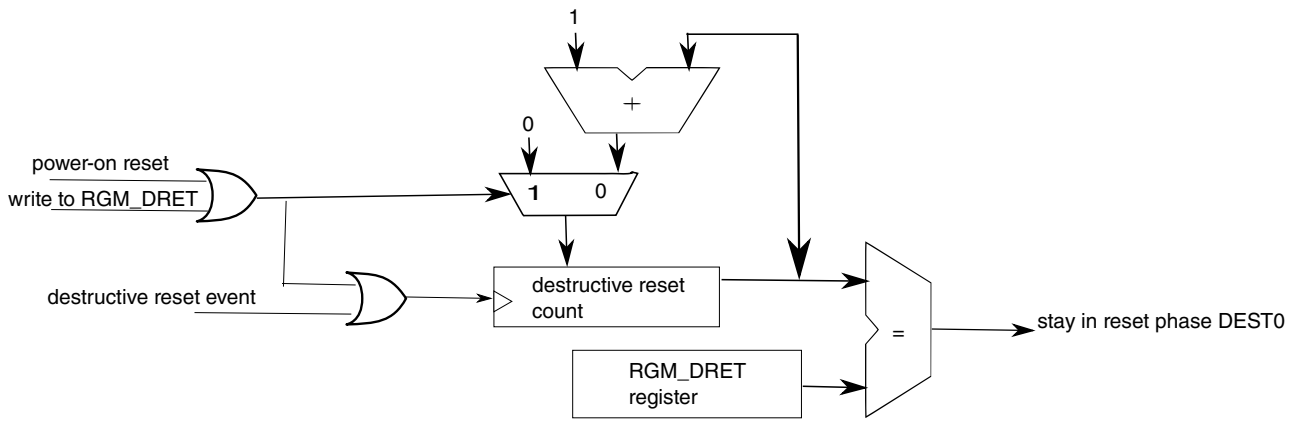


Figure 140. Destructive reset escalation counter

NOTE

Destructive counter increments for each reset source for which escalation is enabled. For details, see "Reset sources" table in the Reset chapter.

32.8 MC_RGM register descriptions

32.8.1 MC RGM Register Map memory map

MC_RGM base address: 4028_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Destructive Event Status Register (DES)	32	RW	0000_0001h
8h	Functional /External Reset Status Register (FES)	32	RW	0000_0000h
Ch	Functional Event Reset Disable Register (FERD)	32	RW	0000_0000h
10h	Functional Bidirectional Reset Enable Register (FBRE)	32	RW	0000_0000h
14h	Functional Reset Escalation Counter Register (FREC)	32	RW	0000_0000h
18h	Functional Reset Escalation Threshold Register (FRET)	32	RW	0000_000Fh
1Ch	Destructive Reset Escalation Threshold Register (DRET)	32	RW	0000_0000h
20h	External Reset Control Register (ERCTRL)	32	RW	0000_0000h
24h	Reset During Standby Status Register (RDSS)	32	RW	0000_0000h
28h	Functional Reset Entry Timeout Control Register (FRENTC)	32	RW	0000_0001h
2Ch	Low Power Debug Control Register (LPDEBUG)	32	R	0000_0000h

32.8.2 Destructive Event Status Register (DES)

Offset

Register	Offset
DES	0h

Function

This register contains the status of the 'destructive' reset sources. This register can be accessed as read/write in supervisor mode and read-only in user mode. Register bits are cleared on write '1'. This register is reset only on power-on reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	DEBUG_D...	SW_DEST	0	0	0	0	0	0	0	0	0	0	HSE_SNV...	HSE_TMP...	0
W		W1C	W1C											W1C	W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SYS_DIV...	HSE_CLK...	0	AIPS_PL...	0	CORE_CL...	PLL_LOL	FXOSC_F...	0	MC_RGM_...	0	STCU_URF	FCCU_FTR	0	0	F_POR
W	W1C	W1C		W1C		W1C	W1C	W1C		W1C		W1C	W1C			W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31 —	Reserved
30 DEBUG_DEST	Flag for 'Destructive' Reset DEBUG_DEST 0b - 'Destructive' reset event DEBUG_DEST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event DEBUG_DEST has occurred.
29 SW_DEST	Flag for 'Destructive' Reset SW_DEST 0b - 'Destructive' reset event SW_DEST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event SW_DEST has occurred.
28 —	Reserved
27	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 HSE_SNVS_RST	Flag for 'Destructive' Reset HSE_SNVS_RST 0b - 'Destructive' reset event HSE_SNVS_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event HSE_SNVS_RST has occurred.
17 HSE_TMPR_RST	Flag for 'Destructive' Reset HSE_TMPR_RST 0b - 'Destructive' reset event HSE_TMPR_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event HSE_TMPR_RST has occurred.
16 —	Reserved
15 SYS_DIV_FAIL	Flag for 'Destructive' Reset SYS_DIV_FAIL 0b - 'Destructive' reset event SYS_DIV_FAIL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event SYS_DIV_FAIL has occurred.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 HSE_CLK_FAIL	Flag for 'Destructive' Reset HSE_CLK_FAIL 0b - 'Destructive' reset event HSE_CLK_FAIL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event HSE_CLK_FAIL has occurred.
13 —	Reserved
12 AIPS_PLAT_CLK_FAIL	Flag for 'Destructive' Reset AIPS_PLAT_CLK_FAIL 0b - 'Destructive' reset event AIPS_PLAT_CLK_FAIL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event AIPS_PLAT_CLK_FAIL has occurred.
11 —	Reserved
10 CORE_CLK_FAIL	Flag for 'Destructive' Reset CORE_CLK_FAIL 0b - 'Destructive' reset event CORE_CLK_FAIL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event CORE_CLK_FAIL has occurred.
9 PLL_LOL	Flag for 'Destructive' Reset PLL_LOL 0b - 'Destructive' reset event PLL_LOL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event PLL_LOL has occurred.
8 FXOSC_FAIL	Flag for 'Destructive' Reset FXOSC_FAIL 0b - 'Destructive' reset event FXOSC_FAIL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event FXOSC_FAIL has occurred.
7 —	Reserved
6 MC_RGM_FRE	Flag for 'Destructive' Reset MC_RGM_FRE 0b - 'Destructive' reset event MC_RGM_FRE has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event MC_RGM_FRE has occurred.
5 —	Reserved
4	Flag for 'Destructive' Reset STCU_URF

Table continues on the next page...

Table continued from the previous page...

Field	Function
STCU_URF	0b - 'Destructive' reset event STCU_URF has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event STCU_URF has occurred.
3 FCCU_FTR	Flag for 'Destructive' Reset FCCU_FTR 0b - 'Destructive' reset event FCCU_FTR has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event FCCU_FTR has occurred.
2 —	Reserved
1 —	Reserved
0 F_POR	Flag for power-on reset **NOTE** If this field is set, ignore all the fields of Destructive Event Status Register (DES) and Functional /External Reset Status Register (FES) registers at power up. 0b - No power-on event has occurred since the last clear. 1b - A power-on event has occurred.

32.8.3 Functional /External Reset Status Register (FES)

Offset

Register	Offset
FES	8h

Function

This register contains the status of the 'functional' and external reset sources. This register can be accessed as read/write in supervisor mode and read-only in user mode. Register bits are cleared on write '1' if the triggering event has already been cleared at the source. This register is reset only on power-on reset.

NOTE

If functional reset escalation to destructive reset is disabled, then the status of this register must be ignored if the fields of Destructive Event Status Register (DES) other than DES[F_POR] are set.

NOTE

If functional reset escalation to destructive reset is enabled and if the bit fields of Destructive Event Status Register (DES), other than DES[F_POR], are set, then based on these fields user should check if the cause of the destructive reset was due to functional reset escalation or if it was triggered directly by a destructive reset source.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	DEBUG_F...	SW_FUNC	0	0	0	0	0	0	0	0	HSE_BOO...	0	0	0	HSE_SWT...
W		W1C	W1C									W1C				W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	JTAG_RST	0	SWT1_RST	SWT0_RST	0	ST_DONE	FCCU_RST	0	0	F_EXR
W							W1C		W1C	W1C		W1C	W1C			W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 DEBUG_FUNC	Flag for 'Functional' Reset DEBUG_FUNC 0b - 'Functional' reset event DEBUG_FUNC has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event DEBUG_FUNC has occurred.
29 SW_FUNC	Flag for 'Functional' Reset SW_FUNC 0b - 'Functional' reset event SW_FUNC has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event SW_FUNC has occurred.
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
22 —	Reserved
21 —	Reserved
20 HSE_BOOT_RST	Flag for 'Functional' Reset HSE_BOOT_RST 0b - 'Functional' reset event HSE_BOOT_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event HSE_BOOT_RST has occurred.
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 HSE_SWT_RST	Flag for 'Functional' Reset HSE_SWT_RST 0b - 'Functional' reset event HSE_SWT_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event HSE_SWT_RST has occurred.
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 JTAG_RST	Flag for 'Functional' Reset JTAG_RST 0b - 'Functional' reset event JTAG_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event JTAG_RST has occurred.
8 —	Reserved
7 SWT1_RST	Flag for 'Functional' Reset SWT1_RST 0b - 'Functional' reset event SWT1_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event SWT1_RST has occurred.
6 SWT0_RST	Flag for 'Functional' Reset SWT0_RST 0b - 'Functional' reset event SWT0_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event SWT0_RST has occurred.
5 —	Reserved
4 ST_DONE	Flag for 'Functional' Reset ST_DONE 0b - 'Functional' reset event ST_DONE has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event ST_DONE has occurred.
3 FCCU_RST	Flag for 'Functional' Reset FCCU_RST 0b - 'Functional' reset event FCCU_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event FCCU_RST has occurred.
2 —	Reserved
1 —	Reserved
0 F_EXR	Flag for External Reset External reset is a source of destructive reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No external reset event has occurred since either the last clear or the last power-on reset assertion. 1b - An external reset event has occurred.

32.8.4 Functional Event Reset Disable Register (FERD)

Offset

Register	Offset
FERD	Ch

Function

This register provides dedicated fields to disable functional reset sources. When any of these reset sources are disabled, the associated functional event is demoted to trigger an interrupt request. This register can be accessed as read/write in supervisor mode and read-only in user mode. Each byte can be written to only once after a destructive or power-on reset and this register is reset only on power-on and any destructive reset.

NOTE

It is important to clear the Functional /External Reset Status Register (FES) before writing 1 to any of the fields in this register. Otherwise a interrupt request may occur.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	D_JTAG_...	0	D_SWT1_...	D_SWT0_...	0	0	D_FCCU_...	0	0	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30	DEBUG_FUNC Disable Control

Table continues on the next page...

Table continued from the previous page...

Field	Function
D_DEBUG_FUNC	0b - Functional reset event DEBUG_FUNC triggers a reset sequence. 1b - Functional reset event DEBUG_FUNC generates an interrupt request.
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 D_JTAG_RST	JTAG_RST Disable Control 0b - Functional reset event JTAG_RST triggers a reset sequence. 1b - Functional reset event JTAG_RST generates an interrupt request.
8 —	Reserved
7 D_SWT1_RST	SWT1_RST Disable Control 0b - Functional reset event SWT1_RST triggers a reset sequence. 1b - Functional reset event SWT1_RST generates an interrupt request.
6 D_SWT0_RST	SWT0_RST Disable Control 0b - Functional reset event SWT0_RST triggers a reset sequence. 1b - Functional reset event SWT0_RST generates an interrupt request.
5 —	Reserved
4 —	Reserved
3 D_FCCU_RST	FCCU_RST Disable Control

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Functional reset event FCCU_RST triggers a reset sequence. 1b - Functional reset event FCCU_RST generates an interrupt request.
2 —	Reserved
1 —	Reserved
0 —	Reserved

32.8.5 Functional Bidirectional Reset Enable Register (FBRE)

Offset

Register	Offset
FBRE	10h

Function

This register enables the generation of an external reset on 'functional' reset.This register can be accessed as read/write in supervisor modeand read-only in user mode.This register is reset on power-on and any 'destructive' reset.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	BE_JTAG...	0	BE_SWT1...	BE_SWT0...	0	BE_ST_D...	BE_FCCU...	0	0	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 BE_DEBUG_FUNC	Bidirectional Reset Enables for 'Functional' Reset DEBUG_FUNC 0b - External reset pin is asserted on a 'Functional' reset DEBUG_FUNC event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset DEBUG_FUNC event.
29 BE_SW_FUNC	Bidirectional Reset Enables for 'Functional' Reset SW_FUNC 0b - External reset pin is asserted on a 'Functional' reset SW_FUNC event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset SW_FUNC event.
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 BE_HSE_BOOT_RST	Bidirectional Reset Enables for 'Functional' Reset HSE_BOOT_RST 0b - External reset pin is asserted on a 'Functional' reset HSE_BOOT_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset HSE_BOOT_RST event.
19	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
18 —	Reserved
17 —	Reserved
16 BE_HSE_SWT_RST	Bidirectional Reset Enables for 'Functional' Reset HSE_SWT_RST 0b - External reset pin is asserted on a 'Functional' reset HSE_SWT_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset HSE_SWT_RST event.
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 BE_JTAG_RST	Bidirectional Reset Enables for 'Functional' Reset JTAG_RST 0b - External reset pin is asserted on a 'Functional' reset JTAG_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset JTAG_RST event.
8 —	Reserved
7 BE_SWT1_RST	Bidirectional Reset Enables for 'Functional' Reset SWT1_RST 0b - External reset pin is asserted on a 'Functional' reset SWT1_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset SWT1_RST event.
6	Bidirectional Reset Enables for 'Functional' Reset SWT0_RST

Table continues on the next page...

Table continued from the previous page...

Field	Function
BE_SWT0_RST	0b - External reset pin is asserted on a 'Functional' reset SWT0_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset SWT0_RST event.
5 —	Reserved
4 BE_ST_DONE	Bidirectional Reset Enables for 'Functional' Reset ST_DONE 0b - External reset pin is asserted on a 'Functional' reset ST_DONE event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset ST_DONE event.
3 BE_FCCU_RST	Bidirectional Reset Enables for 'Functional' Reset FCCU_RST 0b - External reset pin is asserted on a 'Functional' reset FCCU_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset FCCU_RST event.
2 —	Reserved
1 —	Reserved
0 —	Reserved

32.8.6 Functional Reset Escalation Counter Register (FREC)

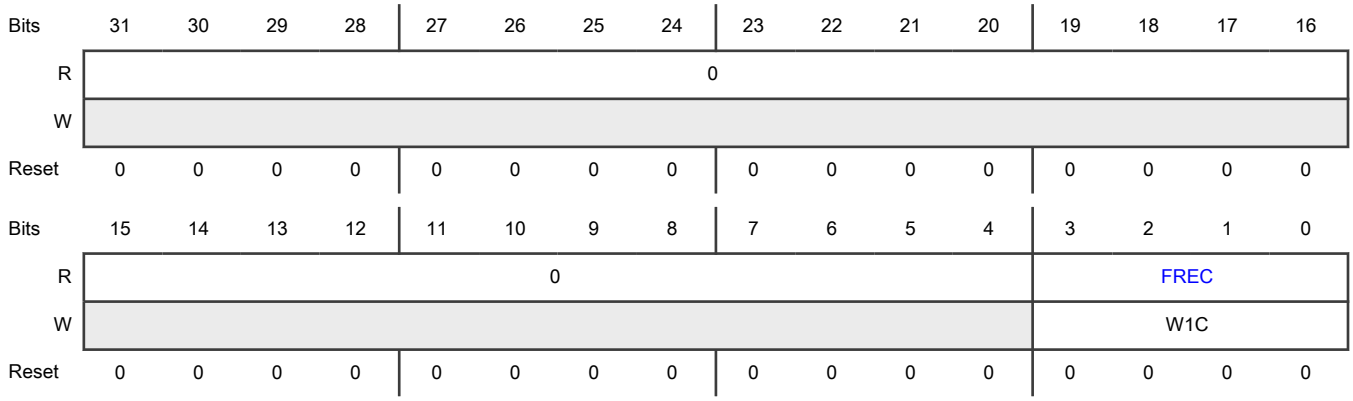
Offset

Register	Offset
FREC	14h

Function

This register provides the current value of functional reset escalation counter. It can be accessed in read/write, either in supervisor mode. It can be accessed in read in the user mode. This register is reset by power-on reset, destructive reset, when you reconfigure the FREC field to Fh and when you write any value to the Functional Reset Escalation Threshold Register (FRET) register.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 FREC	Functional' Reset Escalation Counter This bit field provides the value of functional reset escalation counter.

32.8.7 Functional Reset Escalation Threshold Register (FRET)

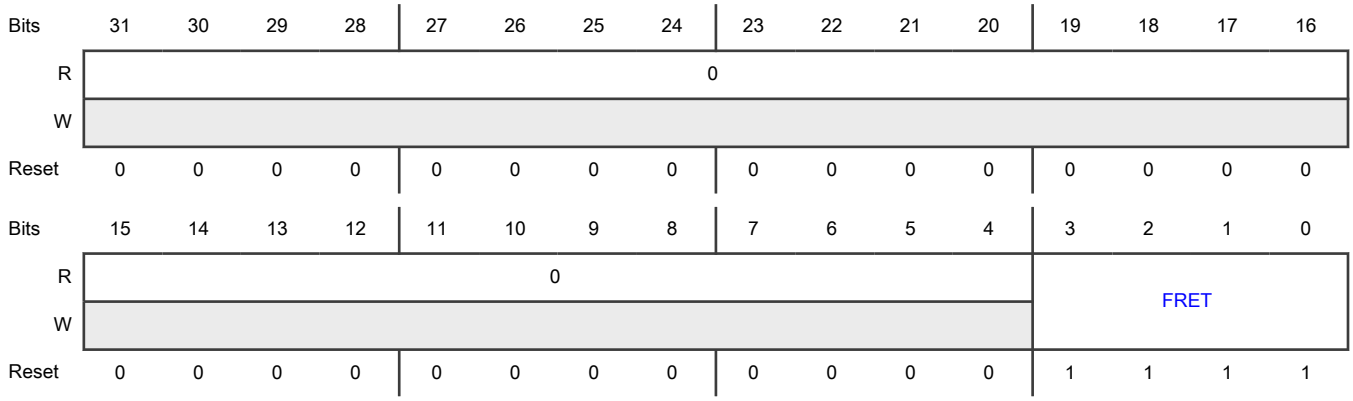
Offset

Register	Offset
FRET	18h

Function

This register sets the threshold for 'functional' reset escalation to a 'destructive' reset. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode. Writing a non-zero value to the FRET field enables the 'functional' reset escalation function. Writing any value to this register resets the 'functional' reset escalation counter. See Functional reset escalation for details on the 'functional' reset escalation function. This register is reset on power-on and any 'destructive' reset.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 FRET	'Functional' Reset Escalation Threshold If the value of this field is 0, the 'functional' reset escalation function is disabled. Any other value is the number of 'functional' resets that causes a 'destructive' reset.

32.8.8 Destructive Reset Escalation Threshold Register (DRET)

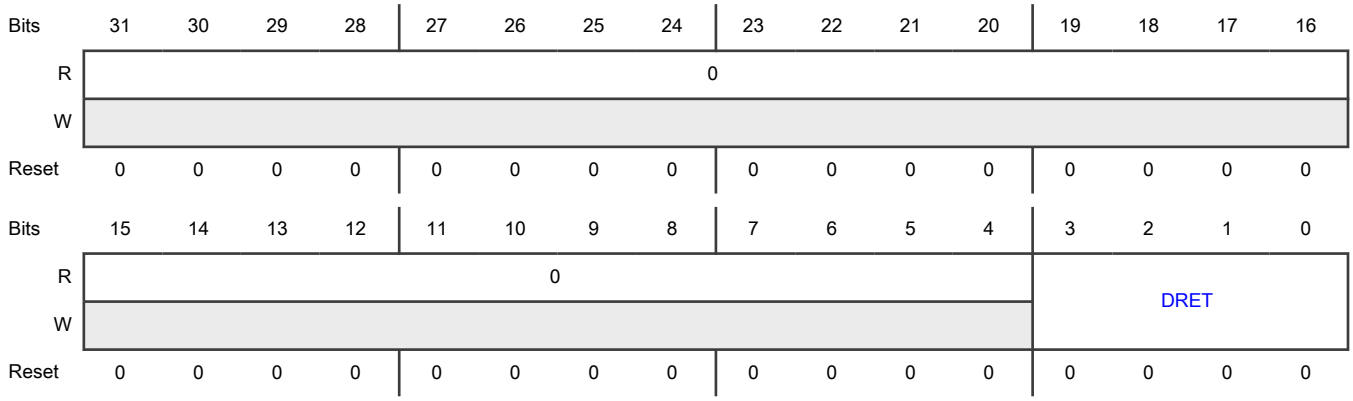
Offset

Register	Offset
DRET	1Ch

Function

This register sets the threshold for 'destructive' reset escalation to keeping the chip in the reset state until the next power-on reset triggers a new reset sequence. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode. Writing a non-zero value to the DRET field enables the 'destructive' reset escalation function. Writing any value to this register resets the 'destructive' reset counter. See Destructive reset escalation for details on the 'destructive' reset escalation function. This register is reset only on power-on reset.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 DRET	'Destructive' Reset Escalation Threshold If the value of this field is 0, the 'destructive' reset escalation function is disabled. Any other value is the number of 'destructive' resets which keeps the chip in the reset state until the next power-on reset triggers a new reset sequence.

32.8.9 External Reset Control Register (ERCTRL)

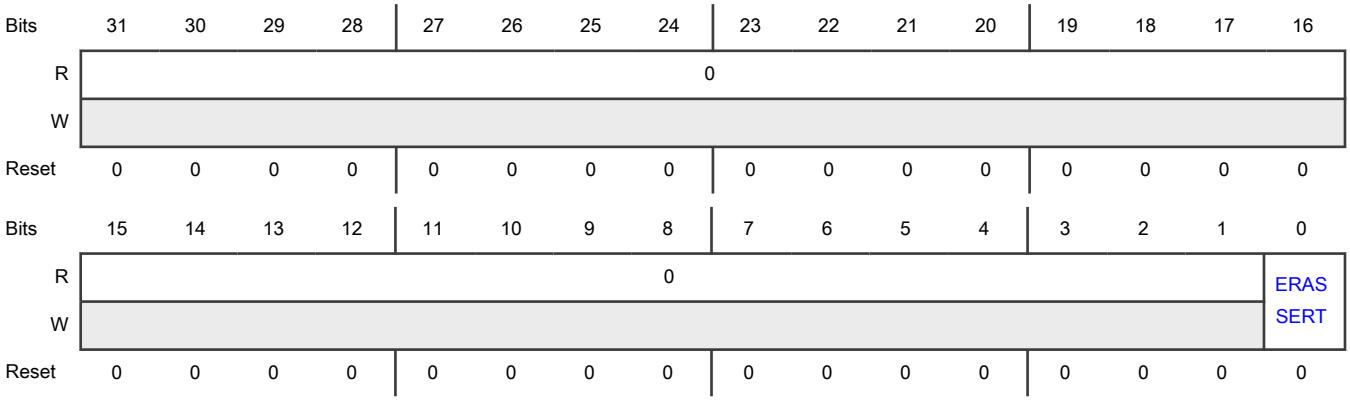
Offset

Register	Offset
ERCTRL	20h

Function

This register allows software to control the assertion of External reset pin. It can be accessed in read/write, in supervisor mode. It can be accessed in read-only in the user mode. This register is reset on all resets.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 ERASSERT	ERASSERT **NOTE** Setting ERASSERT to 1b also safe/pad states most of the chip's pins. See the IOMUX table/spreadsheet for each pin's safe/pad state value. Software must use the ERASSERT bit for this purpose only as part of the main reset domain self-test entry procedure. Using it at any other time may result in unpredictable system behavior. 0b - No change 1b - External reset is asserted

32.8.10 Reset During Standby Status Register (RDSS)

Offset

Register	Offset
RDSS	24h

Function

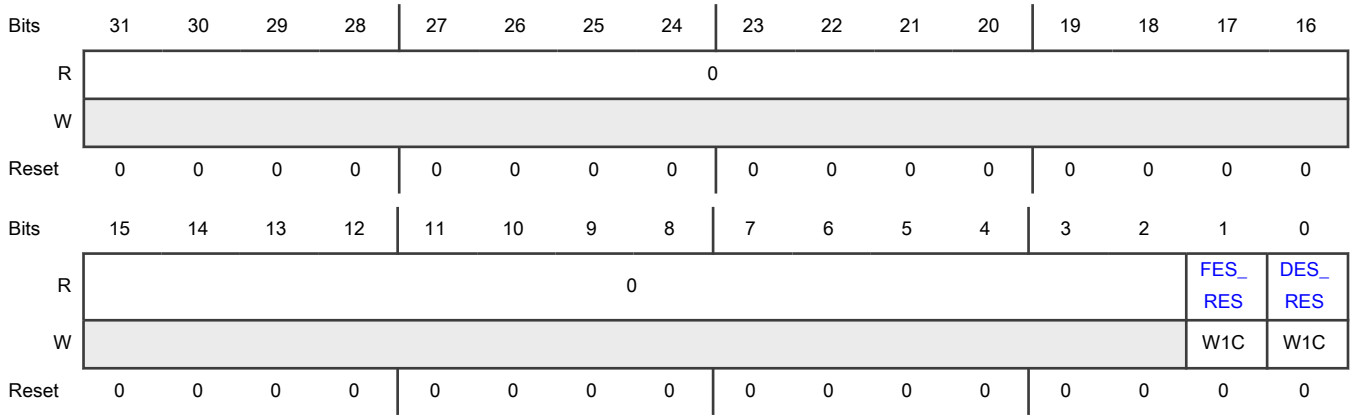
This register provides status of whether a reset event occurred during standby mode. Register bits are cleared on write '1'. This register is reset only on power-on reset.

NOTE

On exiting a reset sequence after standby exit, the software must perform a read operation on MC_ME[PREV_MODE] and RDSS register. If any bit of the RDSS register is set, the software must ignore the status reported by MC_ME[PREV_MODE] register otherwise the status of MC_ME[PREV_MODE] register reports the device status.

If MC_ME indicates last mode as RESET, then perform a reset exit in software else a standby exit.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 FES_RES	FES_RES 0b - No functional reset event occurred during standby mode. 1b - Functional reset event occurred during standby mode.
0 DES_RES	DES_RES 0b - No destructive reset event occurred during standby mode. 1b - Destructive reset event occurred during standby mode.

32.8.11 Functional Reset Entry Timeout Control Register (FRENTC)

Offset

Register	Offset
FRENTC	28h

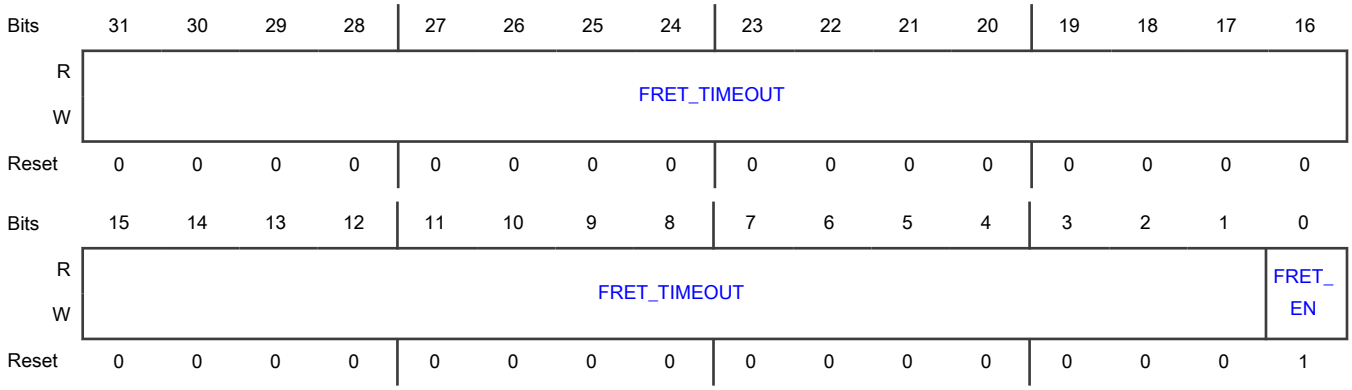
Function

This register provides control for functional reset entry timeout counter. This register is reset on power-on and destructive reset events. This is a word accessible register. Byte accesses are not allowed on this register.

NOTE

This field should be set a value greater than one.

Diagram



Fields

Field	Function
31-1 FRET_TIMEOUT	Functional Reset Entry Timer Value This field allows to override the timeout duration of the functional reset entry timer. The counter value is in terms of the Fast IRC clock cycles (typical clock frequency of 48 MHz). A non zero value of this field would indicate the timer value to be used during the functional reset entry sequence.
0 FRET_EN	Functional Reset Entry Timer Enable/Disable 0b - Functional reset entry timer is disabled. 1b - Functional reset entry timer is enabled

32.8.12 Low Power Debug Control Register (LPDEBUG)

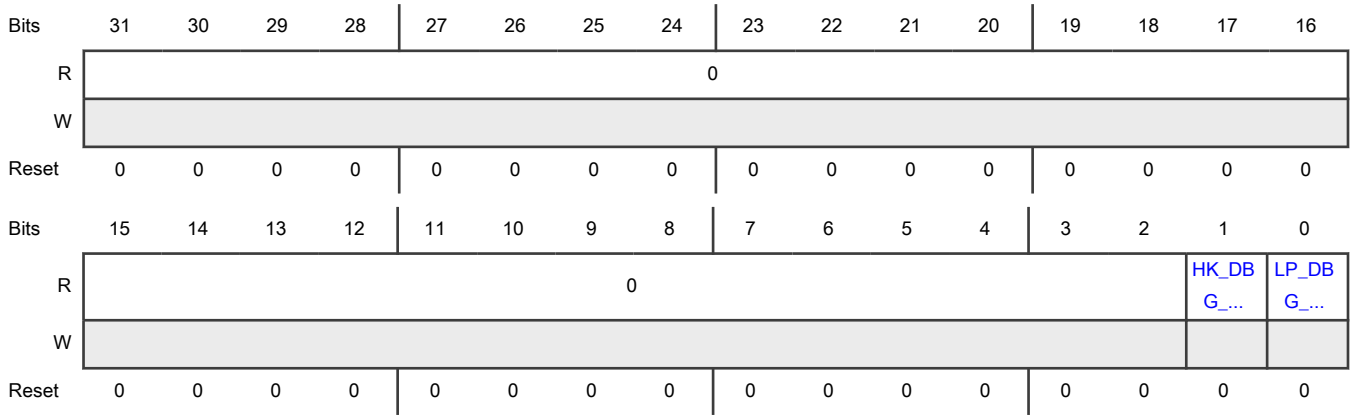
Offset

Register	Offset
LPDEBUG	2Ch

Function

This register provides control when the low-power debug is enabled and you need to wait for debug data collection before moving into standby. This register is reset on power on reset.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 HK_DBG_EN	House Keeping Debugger Enable/Disable Control 0b - House Keeping debugger disabled 1b - Debug data collection enabled for house keeping debugger
0 LP_DBG_EN	Low-Power Debug Enable/Disable Control 0b - Low-power debug disabled 1b - Debug data collection enabled before entering low-power debug

Chapter 33
Power-on Reset Watchdog (POR_WDG)

33.1 Introduction

POR_WDG monitors the chip for stuck or hang scenarios when in reset and standby sequences. It generates a chip power-on-reset event to recover the chip in case it remains stuck in reset or standby entry/exit duration for a pre-configured period of time.

POR_WDG consists of a watchdog counter with a configurable threshold. After the threshold is reached POR_WDG generates a chip power-on-reset event.

SIRC clocks POR_WDG.

33.2 Features

- Counter with four possible configurable threshold limits
- Associate register interfaces to capture the status of input signals
- Monitors reset, standby entry, and standby exit

33.3 Configurations

POR_WDG is enabled by default out of reset to actively monitor the chip reset sequence with the default timeout threshold duration. The POR_WDG configuration controls support provisions for enabling or disabling POR_WDG and the counter threshold, as described in the following sections.

33.3.1 Enable/disable configurations

By default POR_WDG is enabled for reset sequence monitoring, as well as standby entry/exit sequence monitoring for stuck scenarios. The following table describes the controls to enable or disable POR_WDG operation.

Table 181. POR_WDG enable/disable control

POR_WDG monitoring	Default configuration	Enable/disable
Reset monitoring	Enabled	dcf_client_utest_misc[16] Enables or disables POR_WDG for reset monitoring. 0: Disable 1: Enable See the DCF clients file attached to this document for details.
Standby entry/exit monitoring	Disabled	DCM's DCMRWP1[8] Enables or disables POR_WDG for standby entry/exit monitoring. 0: Enable 1: Disable See the DCM chapter for details.

33.3.2 Timeout configurations

POR_WDG supports four configurable counter threshold levels for its counter, as listed in the table below. The default POR_WDG timeout is 25 ms.

Table 182. POR_WDG timeout configuration

DCM's DCMRWP1[10]	DCM's DCMRWP1[9]	POR_WDG timeout
0	0	6.25 ms
0	1	12.50 ms
1	0	25.00 ms
1	1	50.00 ms

33.3.3 Event and status registers

POR_WDG provides indication of a POR_WDG event along with the status of the chip at the point when the POR_WDG counter overflows.

POR_WDG event status register: A POR_WDG event is captured in DCM status register DCMROPP4[0].

POR_WDG status registers: The status of the chip at the POR_WDG event is captured in DCM status registers DCMROPP1–DCMROPP3. See the DCM chapter for details.

DCM's DCMROPP*n* are reset on a POR caused by PMC and are unaffected by a POR_WDG reset.

33.4 Reset sequence monitoring

33.4.1 Introduction

At power-up POR_WDG is enabled for reset monitoring (DCF record in UTEST Misc can bypass POR_WDG, as described in Table 181).

On every reset event, POR_WDG starts monitoring the reset sequence (including the functional reset entry sequence, in case of functional reset event).

If the chip does not exit the reset sequence within the POR_WDG timeout threshold, then POR_WDG initiates a POR sequence to recover the chip from a stuck or hung scenario. You can view the POR_WDG status in DCM's DCMROPPn registers to obtain:

- Diagnostic status
- Chip details in the event of a POR_WDG reset

33.4.2 Inactive windows or bypass operation

POR_WDG monitoring of the reset sequence is inactive in the following conditions:

- POR_WDG is disabled for reset monitoring by chip configuration in UTEST via dcf_client_utest_misc[16]. See the DCF clients file attached to this document for details.
- MC_RGM destructive reset escalation (ensures that the chip stays in reset).
- Extend pin reset by keeping it pressed externally.
- During selftest, when MC_RGM.RGM_ERCTRL[ERASSERT] is configured as 1.

33.4.3 Windows of operation

POR_WDG monitors the reset sequence in the following scenarios:

- Reset pin is active (when chip is in reset, including the functional reset entry sequence).
- Standby domain functional reset is asserted (when reset pad is pulled up but functional reset is asserted).
- RUN domain destructive reset is asserted but flash memory is not in Low-Power mode (when RUN domain destructive reset fires unexpectedly in Run mode).

- RUN domain functional reset is asserted but flash memory is not in Low-Power mode (when RUN domain functional reset fires unexpectedly in RUN mode).

33.5 Standby entry/exit sequence monitoring

33.5.1 Introduction

On every low-power entry or exit event, POR_WDG starts monitoring the low-power sequence (standby entry and standby exit sequence).

If the chip does not exit the standby entry or exit sequence within the POR_WDG timeout threshold, POR_WDG initiates a POR sequence to recover the chip from the stuck or hung scenario. You can then check the POR_WDG status registers in DCM to check the chip status when POR_WDG raised the POR to the chip.

33.5.2 Inactive windows/bypass operation

You can configure DCM's DCMRWP1[8] to disable POR_WDG for standby entry and exit sequence monitoring.

33.5.3 Windows of operation

POR_WDG monitors the standby sequence (standby entry and standby exit) in these scenarios:

- Standby entry sequence monitoring: Monitoring from main core WFI execution until PMC acknowledges Low-Power mode entry.
- Standby exit sequence monitoring: Monitoring from wakeup event until the RUN domain reset recovery.

33.6 Glossary

WFI	Wait for Interrupt

Chapter 34
Security Overview

34.1 Introduction

This chip has a comprehensive set of customer-configurable security features designed to protect code and data from unauthorized access. The content of this section is for non-secure operation only. Contact your NXP representative for details if you need secure operation.

34.2 Security features

MWCT2xxxS:

- Exceeds the EVITA- full [5] functionality and offers high performance for edge nodes applications.
- Bases its chip censorship on the life cycle model. Access to chip code and data becomes progressively more restricted as the chip matures through a defined set of life cycle steps.
- Offers these memory security features:
  - NVM censorship support
  - Debug password protection
  - OTP flash memory areas
- Supports a unique ID—The chip has a unique ID stored in an OTP flash memory area. Any core on the chip can read this unique ID.
- Includes HSE_B, which is a dedicated security system providing:
  - A processor core
  - Dedicated SRAM
  - Symmetric Hardware Accelerator
  - Asymmetric Hardware Accelerator
  - True Random Number Generator (TRNG)
  - Pseudo Random Number Generator (PRNG)
  - Exclusive access to secure areas of the chip's flash memory
  - OTA : In Field Secure Code/Data updates

NOTE

Refer Security Reference Manual and Firmware Reference Manual for the details on OTA.

Also, HSE_B runs NXP firmware independently from the main chip processor cores and can implement advanced security and monitoring functions.

[5] EVITA was a European research project existing from 2008 to 2011. There is no standard or a specification to test compliancy. However, HSE_B meets the common expectations often described in the industry as EVITA Full and exceeds them in these cases:
- HSE_B supports up to AES-256 instead of AES-128
- HSE_B supports up to ECC-521 instead of ECC-256
- HSE_B supports SHA-2/Miyaguchi-Preneel instead of Whirlpool

- Includes AES_ACCEL, which provides an independent DMA-controlled crypto accelerator with security features. AES_ACCEL provides timeout counters. You can transfer keys from HSE_B to the AES_ACCEL keystore, which can hold 80 keys.
- Supports a secure debugger interface.
- Provides boot modes:
  - Trusted and secure boot support
  - Handshake that supports BAF
- Monitors operating conditions to maximize tampering resistance.
- Includes basic debugger restrictions (on and off via censorship mode).

When HSE_B detects a security failure, it moves the chip to a secure state. The chip secure states comply with the chip safe states. You enter a safe state as reset when there are security errors leading to reset. Moreover, if the chip stays in Run mode and the lifecycle moves to the IN_FIELD phase in reaction to errors, the safety application can continue running without glitches.

34.3 Security information

Chip security feature details are published in the HSE firmware reference manual and HSE service interface manual. Contact your NXP sales representative for more information.

34.4 Glossary

AES	Advanced encryption standard
BAF	Boot assist flash
ECC	Elliptic curve cryptography
EVITA	E-safety vehicle intrusion protected applications
NVM	Nonvolatile memory
OTP	One-time programmable

Chapter 35
Hardware Security Engine (HSE_B)

35.1 HSE subsystem

HSE is a security subsystem. It runs security functions for applications having stringent confidentiality and authenticity requirements. HSE has the following objectives:

- Isolating security-sensitive information (for example, secret keys) from the application (the host)
- Transferring the cryptographic operations from application cores and processing them
- Accelerating cryptographic operations with dedicated coprocessors
- Enforcing security measures on the application, during runtime and system startup

The HSE subsystem is the only master that is unconditionally released from reset after POR. It then releases the CPU subsystems in the host from reset, with the opportunity to apply certain checks beforehand (secure system startup). Based on certain conditions, HSE can also trigger interrupts and reset signals to the host during runtime.

35.1.1 CPU subsystem

The HSE CPU subsystems process the security functions and control system resources to provide security services to the application domain (the host).

35.1.2 Cryptographic accelerators

The HSE subsystem supports the following cryptographic accelerators:

- An AES engine supporting all standard key sizes (128, 192, 256 bits) and various complex ciphering modes (ECB, CBC, CTR, OFB, CFB, CCM, CMAC)
- A hash engine that supports standard several primitives: MD5, SHA-1, SHA-224, SHA-256
- PKC which accelerates RSA and ECC operations (key generation, signature generation, signature verification, ciphering)
  - RSA (1024, 2048, 3072, 4096-bit)
  - ECC over prime numbers
    - BN P256, P638
    - ANSI X9P192 to X9P512
    - Brainpool P160 to P512
    - Sec P128 to P521
    - TU Darmstadt prime Curve 1 to 35
    - Ed25519

The HSE subsystem supports several other cryptographic primitives through the software. See HSE firmware for more information.

35.1.3 Random number generator (RNG)

The RNG in this chip consists of a TRNG and a DRBG. Both are designed to be compliant to the highest strength in security as specified in

- BSI AIS20/31
- NIST SP800-90a,b,c

The TRNG function provides a seed for the DRBG, while the DRBG is available to the host via dedicated random number generation services.

35.1.4 Timers

The HSE subsystem features:

- An independent dedicated system timer
- HSE_STM (apart from chip timer resources), that allows recurring autonomous functions such as runtime memory verification checks
- A watchdog timer to reset the HSE subsystem in case of unexpected runtime failure

35.1.5 Memory resources

See Configuration_GPR memory map for more information on configuration controls related to HSE functions.

35.1.5.1 Secure RAM

Secure RAM refers to RAM area that the HSE subsystem access exclusively.

35.1.5.2 Secure NVM

Secure NVM refers to nonvolatile memory that the HSE subsystem accesses exclusively.

35.2 HSE interface

35.2.1 Messaging unit (MU)

The HSE subsystem has two messaging units:

- HSE_MU0
- HSE_MU1

See the "Messaging Unit (MU)" chapter for more information.

35.2.1.1 Overview

MU is the communication interface between the host and the HSE subsystem. The host uses MU to trigger service requests and to receive service responses. The HSE firmware uses MU to receive service requests, return service responses, and provide a general status of the HSE subsystem.

Each of the two MU instances available in the HSE subsystem has:

- Two sides:
  - MUA: Only the HSE subsystem accesses it.
  - MUB: The host accesses it.

    The registers on one side have corresponding registers on the other side.
- A set of 32-bit readable and writable transmit registers (MUB_TRn), which the host uses to transfer the address of the service descriptor to the HSE firmware. The HSE firmware retrieves the address of the service descriptor in the corresponding registers MUA_RRn.
- A set of 32-bit read-only receive registers (MUB_RRn), which the host uses to retrieve the response to the service requests. The HSE firmware provides the response to service requests in the corresponding registers MUA_TRn.
- Control and status registers to manage MU and the access to transmit and receive registers.

The advantages of using the MU to manage the HSE service requests and responses are:

- Hardware mechanisms are in place on the transmit registers to avoid service request overrun.

- Interrupt signals are available to allow asynchronous management of the requests (avoiding active waiting loops).
- Freedom from interference between different application cores. You can configure each MU instance with specific access restrictions that can be used, for example, to isolate the requests that different masters make (in different MU instances). You configure such access controls using XRDC.

35.3 Debug

35.3.1 HSE subsystem debug

The debugging of the HSE subsystem and associated firmware is restricted to NXP engineering teams.

35.3.2 Host debug

The host debug is either open or protected, depending on the device Lifecycle state. See the 'Life cycle' section in 'Device Configuration Module' chapter for details on device lifecycle advancement and decoding. The details of the device security features is published in a separate document, the Security Reference Manual. Please contact your NXP sales representative for details. See the Debug chapter for details on host/application core debug.

The debug protection consists of locking the debugger access through the JTAG interface until the HSE firmware authenticates the debugger. This authentication is based on ADK/P, a 16-byte region within UTEST used for application core debug. See UTEST memory map in the DCF clients file attached to this document.

The authentication method can be:

- Static: In this case, ADK/P is a password which the debugger provides in plain form.
- Dynamic: In this case, ADK/P is a cryptographic key which the debugger uses to calculate a cryptographic response to a random challenge.

35.4 HSE firmware

Factory supplied firmware that runs in the HSE subsystem controls HSE functionality. It essentially serves the host with a set of security services as described in Table 183.

Table 183. Summary of firmware security services

Service	Summary
Administration	Install, configure, update, and test the HSE firmware
Key management	Available for the application to manage different sets of keys that the HSE firmware manages, for example, cryptographic services
Cryptographic	Provide the application with cryptographic primitives that high level security stacks use in the application
Random number	Generate random streams that can be used in various security protocols
Memory verification	Allow the application to verify different memory areas at startup (after reset) and during runtime
Monotonic counter	Provide the application with a set of monotonic counters that can be read and only incremented
Secure time	Allow the configuration of a secure tick to be signaled to the application

Table 184 provides an overview of services and features that the HSE firmware supports.

Table 184. HSE firmware features

Service	Category	Feature
Cryptography	Ciphers	AES: ECB, CBC, CFB, OFB, CTR, XTS[1] RSAES: PKCS1-v1_5, OAEP
	Message Authentication Code (MAC)	AES: CMAC, XCBC-MAC[1], HMAC[1]
	Hashing	SHA1 SHA224, SHA256, SHA384, SHA512 SHA3_224[1], SHA3_256[1], SHA3_384[1], SHA3_512[1] MD5 Miyaguchi-Preneel Compression
	Authenticated ciphers	AES: CCM, GCM[1]
	Digital signature generation and verification	RSASSA_PSS RSASSA_PKCS1-v1_5 ECDSA – ECC over GF(P) with all prime standard curve supported EdDSA - Ed25519 pre-hashed curve
Key management	Max key sizes	AES: 256 bits RSA: 4096 bits ECC: 521 bits DH: 4096 bits
	Key generation	Permanent and ephemeral RSA and ECC key pair generation
	Key import or export	Plain or encrypted form, with optional authentication tag. SHE key update protocol
	Key derivation	NIST 800-108, PBKDF2, and so on
	Key exchange	ECDH and Classic DH
	Certificate handling	Key Installation from X.509 and CVC certificates Certificate installation for Root of Trust establishment.
Boot and memory verification	Authentication schema	AES CMAC, XCBC-MAC RSA & ECC signatures
	Verification flow	Before application startup (strict secure boot) In parallel to the application startup On the application demand
	Sanctions	No startup (strict secure boot)

Table continues on the next page...

Table 184. HSE firmware features (continued)

Service	Category	Feature
		Chip reset Key usage restrictions
Monotonic counter	Counter management	Incrementing and reading volatile and non-volatile counters
Random number	Deterministic random bit generation	Based on a True Random Number AIS31 Class P2 high and FIPS 140-2 compliant
Secure time	Secure tick	Application interrupts at configurable frequency
Administration services	HSE administration	Firmware installation and update Subsystem configuration and testing

1. Software implementation of cryptographic primitives.

See documents in Table 185 for more information about how to install, configure, and use the HSE firmware that NXP provides.

Table 185. References

Document number	Document title	Description
HSEFWRM	HSE Firmware Reference Manual	Contains details about how to install, configure, and use the HSE subsystem.
HSESIRM	HSE Service Interface Reference Manual	Security firmware API reference for non-AUTOSAR users.

35.5 General purpose registers with write access by HSE core

This is a general purpose register bank that only the HSE core can write. Secure master ID gating allows only the secure core to write in these registers, whereas any application core can read.

35.6 Configuration_GPR register descriptions

35.6.1 Configuration_GPR memory map

This section describes the chip configurations that only the HSE core manages. These constitute control of peripherals, secure size configurations for SRAM and flash memory, flash memory program or erase control, and so on.

NOTE

Write accesses to configuration registers apart from 32-bit accesses might result in unpredictable chip behavior, therefore must not be done.

Configuration_GPR base address: 4039_C000h

Offset	Register	Width (In bits)	Access	Reset value
1Ch	General Purpose Configuration 0 (CONFIG_REG0)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
34h	General Purpose Configuration 6 (CONFIG_REG6)	32	R	0000_0035h
38h	Configuration RAM Protected Region (CONFIG_RAMPR)	32	R	See section
3Ch	Configuration Code Flash Memory Active Block (CONFIG_CFPRL)	32	R	See section
40h	Configuration Code Flash Memory Passive Block (CONFIG_CFPRH)	32	R	See section
44h	Configuration Data Flash Memory Protected Region (CONFIG_DFPR)	32	R	See section
50h	Configuration Program and Erase Lock (CONFIG_PE_LOCK)	32	R	0000_0000h
54h	Configuration RAM Protected Region Alternate (CONFIG_RAMPR_ALT)	32	R	See section
58h	Configuration Code Flash Memory Active Block Alternate (CONFIG_CFPRL_ALT)	32	R	See section
5Ch	Configuration Code Flash Memory Passive Block Alternate (CONFIG_CFPRH_ALT)	32	R	See section
60h	Configuration Data Flash Memory Protected Region Alternate (CONFIG_DFPR_ALT)	32	R	See section
64h	Configuration REG_GPR (CONFIG_REG_GPR)	32	RW	A000_0003h

35.6.2 General Purpose Configuration 0 (CONFIG_REG0)

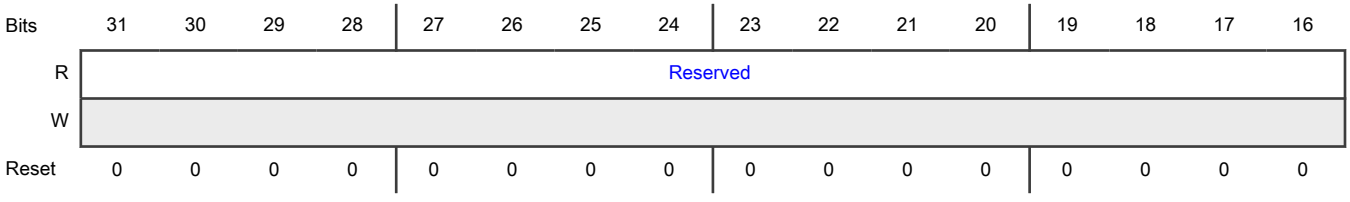
Offset

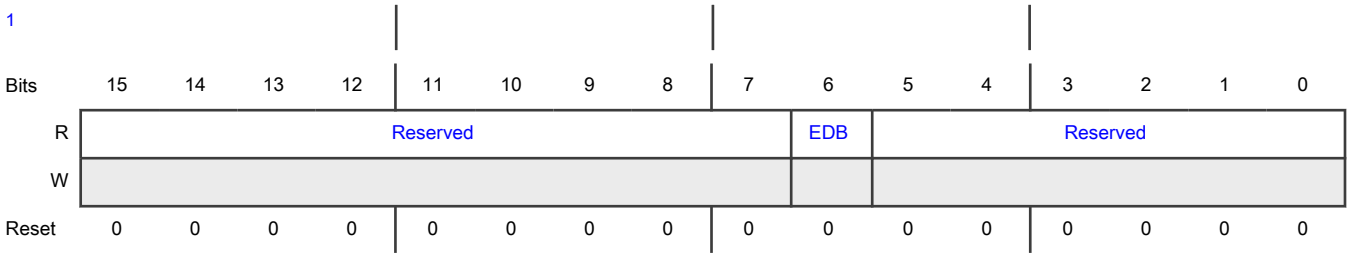
Register	Offset
CONFIG_REG0	1Ch

Function

The EDB field resets on destructive or POR reset events and functional reset has no impact on it.

Diagram





1. If export_control=1, the reset value of EDB is 1. If export_control=0, the reset value is 0.

Fields

Field	Function
31-7 —	Reserved
6 EDB	Hardware Debugger Attached This is a sticky field that clears on destructive reset. 0b - Debugger not connected 1b - Debugger connected
5-0 —	Reserved

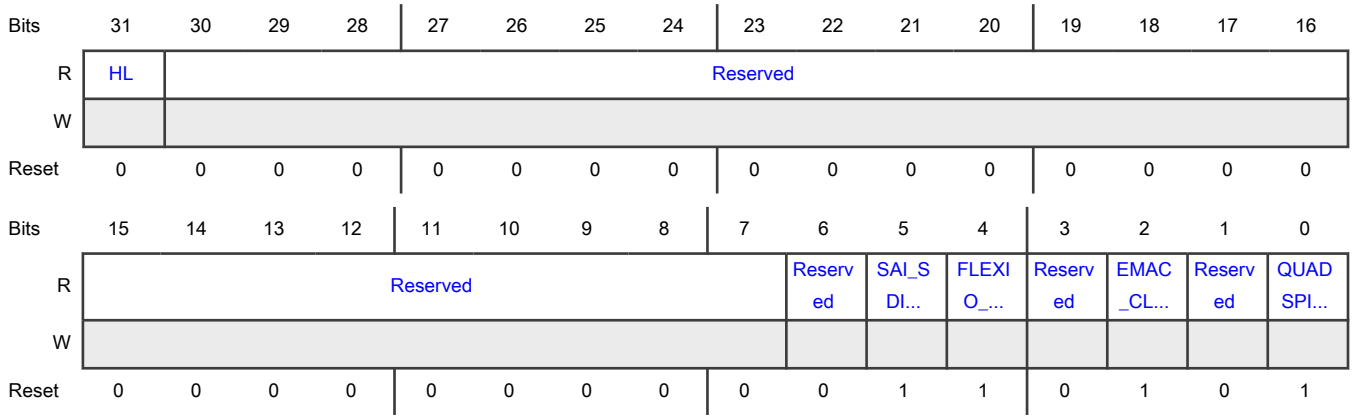
35.6.3 General Purpose Configuration 6 (CONFIG_REG6)

Offset

Register	Offset
CONFIG_REG6	34h

Function
Resets on functional reset.

Diagram



Fields

Field	Function
31 HL	Hard Lock Only functional reset clears this field. 0b - You can write to this register 1b - Register is locked for any write
30-7 —	Reserved
6 —	Reserved
5 SAI_SDID_PCTL	SAI0 and SAI1 clock gating Clock to SAI peripheral is on or off. 0b - Clock is off (gated) 1b - Clock is on
4 FLEXIO_CLOCK_GATE	FlexIO Clock Gating Clock to FlexIO peripheral is on or off. 0b - Clock is off (gated) 1b - Clock is on
3 —	Reserved
2 EMAC_CLOCK_GATE	Ethernet Clock Gating Clock to Ethernet peripheral is on or off.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is off (gated) 1b - Clock is on
1 —	Reserved
0 QUADSPI_SDID_PCTL	QuadSPI Clock Gating Clock to QuadSPI peripheral is on or off. 0b - Clock is off (gated) 1b - Clock is on

35.6.4 Configuration RAM Protected Region (CONFIG_RAMPR)

Offset

Register	Offset
CONFIG_RAMPR	38h

Function

Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HARD_LO...	SOFT_LO...	Reserved										SECURE_SIZE			
W																
Reset [1]	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

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

1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31 HARD_LOCK	Hard Lock This is a sticky field. If you write 1 to it, it remains 1 until next reset. 0b - Write access to this register is allowed 1b - Write access to this register is not allowed until next functional reset
30 SOFT_LOCK	Soft Lock 0b - Write access to this register is allowed 1b - Write access to this register is not allowed
29-21 —	Reserved
20-5 SECURE_SIZE	Secure Size Secure size region (in bytes) for PRAM1. This is 32-byte-aligned to ensure alignment with cache lines.
4-0 —	Reserved

35.6.5 Configuration Code Flash Memory Active Block (CONFIG_CFPRL)

Offset

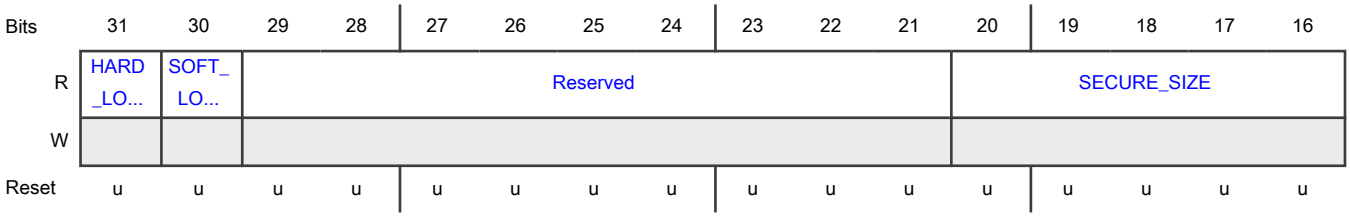
Register	Offset
CONFIG_CFPRL	3Ch

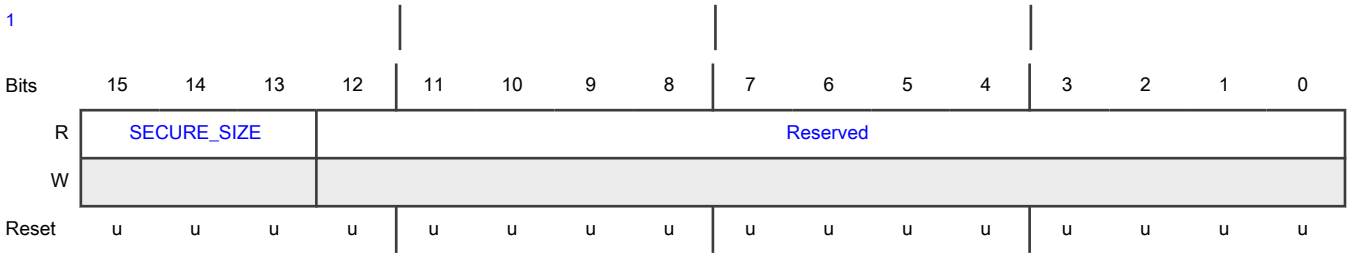
Function
Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram





1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31 HARD_LOCK	Hard Lock After this field is configured it cannot be modified until a reset event. 0b - Write access is allowed 1b - Write access is not allowed until next functional reset
30 SOFT_LOCK	Soft Lock 0b - Write access is allowed 1b - Write access is not allowed
29-21 —	Reserved
20-13 SECURE_SIZE	Secure Size Flash memory active block secure size in bytes to align to 8 KB (sector) aligned write.
12-0 —	Reserved

35.6.6 Configuration Code Flash Memory Passive Block (CONFIG_CFPRH)

Offset

Register	Offset
CONFIG_CFPRH	40h

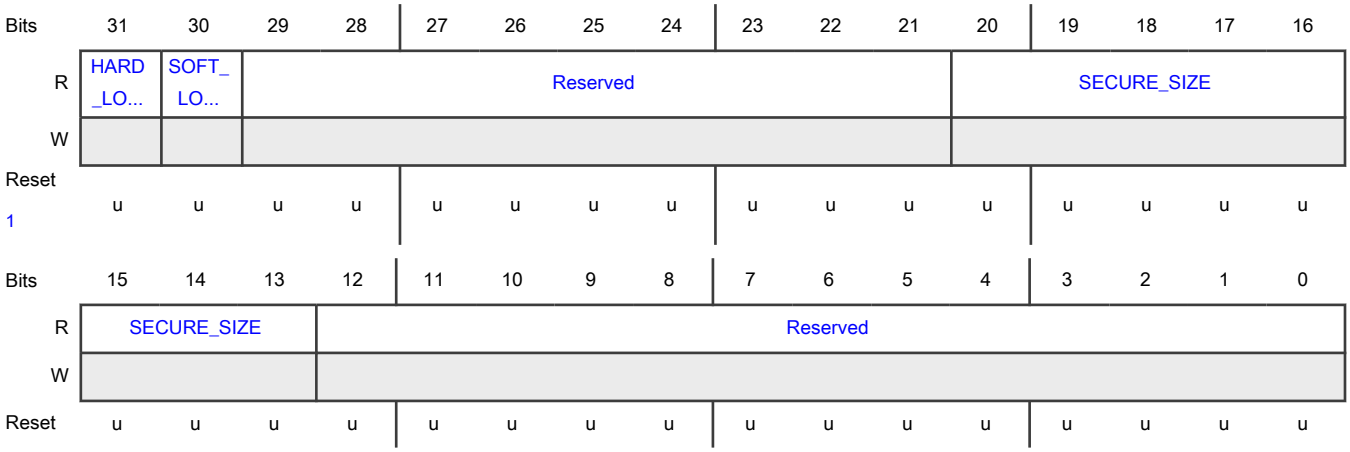
Function

Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram



1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31 HARD_LOCK	Hard Lock After this field is configured, it cannot be modified until a reset event. 0b - Write access is allowed 1b - Write access is not allowed until next functional reset
30 SOFT_LOCK	Soft Lock 0b - Write access is allowed 1b - Write access is not allowed
29-21 —	Reserved
20-13 SECURE_SIZE	Secure Size Secure size region (in bytes) for flash memory passive block for alignment with 8 KB (sector) aligned writes.
12-0 —	Reserved

35.6.7 Configuration Data Flash Memory Protected Region (CONFIG_DFPR)

Offset

Register	Offset
CONFIG_DFPR	44h

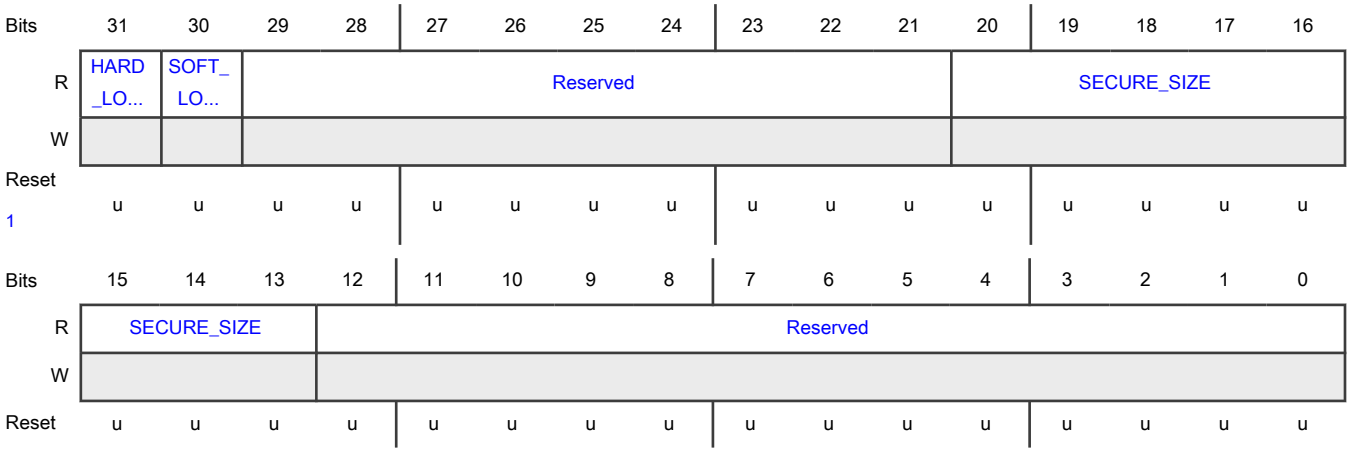
Function

Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram



1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31 HARD_LOCK	Hard Lock After this field is configured, it cannot be modified until a reset event. 0b - Write access is allowed 1b - Write access is not allowed until next functional reset
30 SOFT_LOCK	Soft Lock 0b - Write access is allowed 1b - Write access is not allowed
29-21 —	Reserved
20-13 SECURE_SIZE	Secure Size Secure size region (in bytes) for data flash memory aligned to 8KB (sector) aligned writes.
12-0 —	Reserved

35.6.8 Configuration Program and Erase Lock (CONFIG_PE_LOCK)

Offset

Register	Offset
CONFIG_PE_LOCK	50h

Function

Resets on functional reset.

Diagram

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

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

Fields

Field	Function
31-19 —	Reserved
18 PE_LOCK_UTEST	Program/Erase Lock for UTEST 0b - UTEST lock request is clear 1b - UTEST lock requested
17 —	Reserved
16 PE_LOCK_BLOCK_4	Program/Erase Lock for Block 4 0b - Block 4 lock request is clear 1b - Block 4 lock requested
15 PE_LOCK_BLOCK_3	Program/Erase Lock for Block 3 0b - Block 3 lock request is clear 1b - Block 3 lock requested

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 PE_LOCK_BLOCK_2	Program/Erase Lock for Block 2 0b - Block 2 lock request is clear 1b - Block 2 lock requested
13 PE_LOCK_BLOCK_1	Program/Erase Lock for Block 1 0b - Block 1 lock request is clear 1b - Block 1 lock requested
12 PE_LOCK_BLOCK_0	Program/Erase Lock for Block 0 0b - Block 0 lock request is clear 1b - Block 0 lock requested
11-0 —	Reserved

35.6.9 Configuration RAM Protected Region Alternate (CONFIG_RAMPR_ALT)

Offset

Register	Offset
CONFIG_RAMPR_ALT	54h

Function

Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	INVERT_VALUE_RAMPR															
W																
Reset [1]	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

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

1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31-0 INVERT_VALUE_RAMPR	Invert Value RAMPR Write inverted value of register CONFIG_RAMPR to CONFIG_RAMPR_ALT.

35.6.10 Configuration Code Flash Memory Active Block Alternate (CONFIG_CFPRL_ALT)

Offset

Register	Offset
CONFIG_CFPRL_ALT	58h

Function
Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	INVERT_VALUE_CFPRAB															
W																
Reset [1]	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

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

1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31-0 INVERT_VALUE_CFPRAB	Invert Value CFPRAB Write inverted value of register CONFIG_CFPRL to CONFIG_CFPRL_ALT.

35.6.11 Configuration Code Flash Memory Passive Block Alternate (CONFIG_CFPRH_ALT)

Offset

Register	Offset
CONFIG_CFPRH_ALT	5Ch

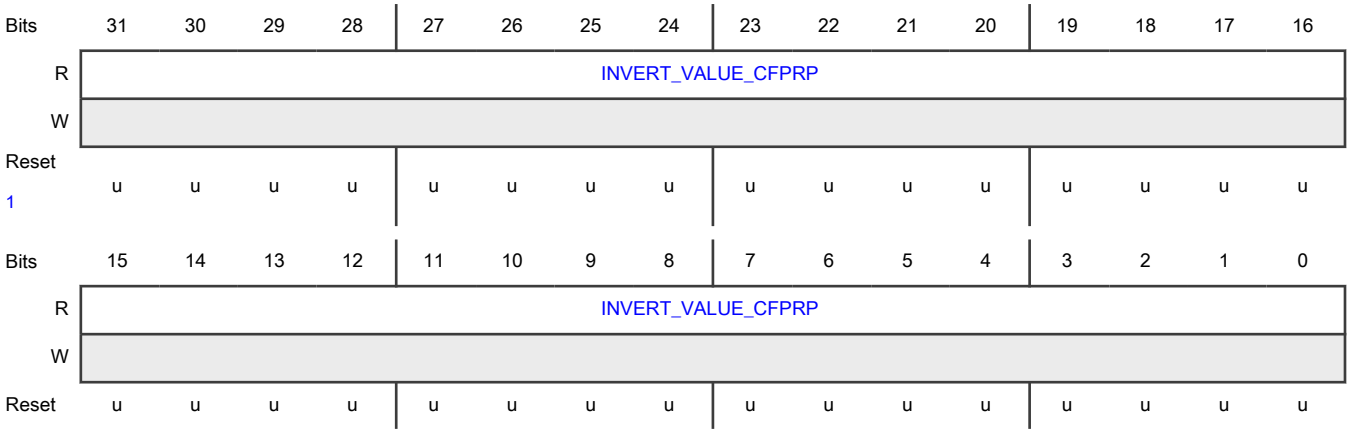
Function

Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram



1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31-0 INVERT_VALUE_CFPRP	Invert Value CFPRP Write inverted value of register CONFIG_CFPRH to CONFIG_CFPRH_ALT.

35.6.12 Configuration Data Flash Memory Protected Region Alternate (CONFIG_DFPR_ALT)

Offset

Register	Offset
CONFIG_DFPR_ALT	60h

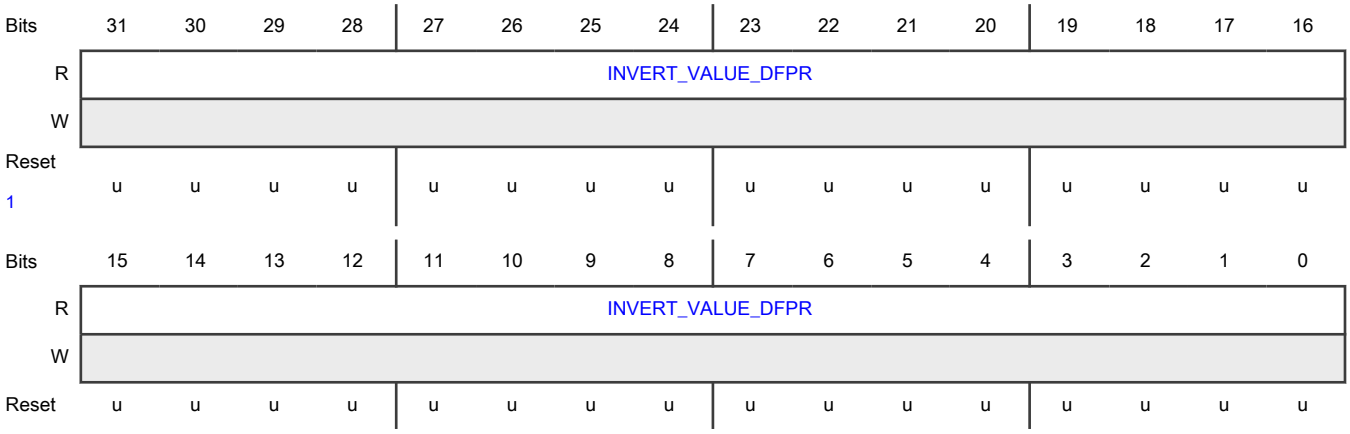
Function

Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram



1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31-0 INVERT_VALUE_DFPR	Invert Value CFPRP Write inverted value of register CONFIG_DFPR to CONFIG_DFPR_ALT.

35.6.13 Configuration REG_GPR (CONFIG_REG_GPR)

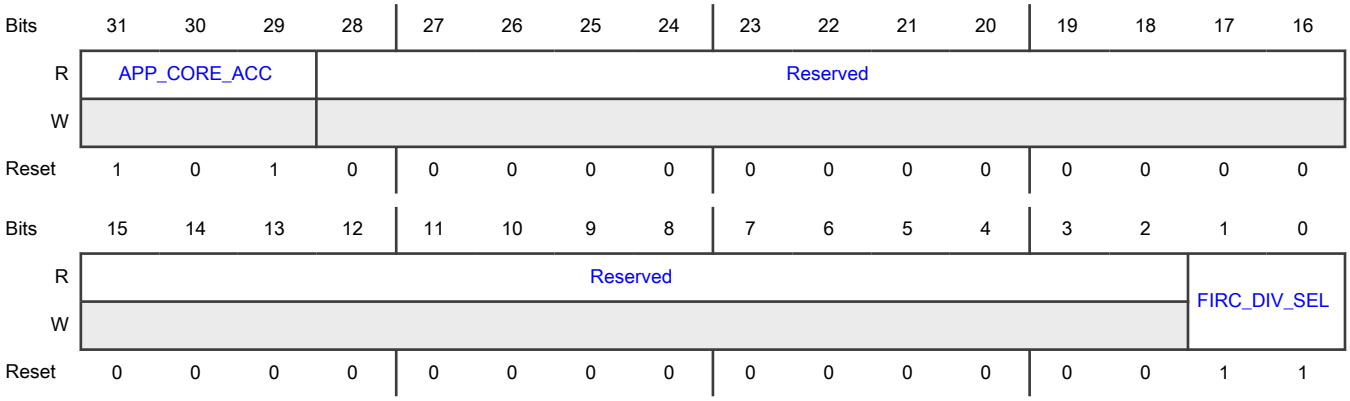
Offset

Register	Offset
CONFIG_REG_GPR	64h

Function

Resets on functional reset.

Diagram



Fields

Field	Function
31-29 APP_CORE_ACC	FIRC Divider **NOTE** While writing to this register, APP_CORE_ACC is RO and should not be changed from 0b101. 101b - Application core can write this field [FIRC_DIV_SEL] All other values - No access to application core
28-2 —	Reserved
1-0 FIRC_DIV_SEL	FIRC Divider Indicates this chip's FIRC clock division factor. 00b - Divided by 2 01b - Divided by 2 10b - Divided by 16 11b - Undivided

35.7 Glossary

ADK/P	Application debug key/password
AES	Advanced encryption standard
CBC	Cipher block chaining
CCM	Counter with CBC MAC (Cipher block chaining message authentication code)
CFB	Cipher feedback mode
CMAC	Cipher-based message authentication code
CTR	Counter-based block cipher mode

CVC	Card verifiable certificates
Classic DH	Classical Diffie–Hellman, a key exchange method
DRBG	Deterministic random bit generator
ECB	Electronic code book
ECC	Elliptic curve cryptography
ECDSA	Elliptic curve digital signature algorithm
ECDH	Elliptic-curve Diffie–Hellman, a key exchange method
EdDSA	Edwards-curve digital signature algorithm
FIPS	Federal information processing standards
MUA	Messaging unit A interface
MUB	Messaging unit B interface
NIST	National institute of standards and technology
OAEP	Optimal asymmetric encryption padding
OFB	Output feedback based block cipher mode
GCM	Galois/counter mode
HMAC	Keyed-hash message authentication code
PKC	Public key cryptographic engine
PKCS1	Public-key cryptography standards. PKCS provides the basic definitions of, and recommendations for implementing the RSA algorithm.
POR	Power-on reset
RNG	Random number generator
RSA	Rivest–Shamir–Adleman (a public key cryptosystem)
SHE	Secure hardware extension
TRNG	True random number generator
XCBC-MAC	Extended ciphertext block chaining MAC
XTS	XEX (XOR encrypt XOR) based tweaked-codebook mode with ciphertext stealing

Chapter 36
Device Configuration Format (DCF) records

36.1 Introduction

DCF configures certain registers of this chip during system boot while the reset signal asserts. An individual DCF record points to an internal register in the chip and the data to be written to that register.

UTEST DCF—The UTEST DCF clients are present within the UTEST region of the flash and programmed during production testing. You may write the other records and program them at the same time the application code is programmed in the flash memory. See the DCF client file attached to this document for the description on the UTEST DCF records.

System boot is a complex process that requires you to initialize the chip properly before releasing reset. Before using the chip, the user writes the application specific code into the flash memory. The user can also update the DCF records from their initial settings as per application requirements, in case if needed.

After power is supplied to an appropriately configured chip, PMC controls the chip, and after the system power supplies reach predefined levels, PMC signals MC_RGM to start the boot sequence. During this sequence, MC_RGM enables DCM to read the chip configuration records and then write the configuration information to the specified registers.

36.2 DCF clients

These are 32-bit wide hardware registers inside a module that receive and store data from a DCF record. This stored data is used to initialize registers and configure features.

DCF clients:

- Are assigned a default value before any DCF records are written.
- May have special writing constraints, such as:
  - Write once.
  - Change from 1 to 0 only.
  - Change from 0 to 1 only.
- May not implement all 32 bits.

36.2.1 Safety features of DCF clients

Depending on the DCF client's role in the chip, the client may be equipped with a safety feature or a combination of these features.

36.2.1.1 Parity

If a DCF client implements parity checking, the client receives a parity bit in addition to its data in the DCF record. During chip operation, the client continuously monitors whether the data it stores matches the parity. The parity scheme used is even parity. So, the number of 1s in the WDATA and parity field needs to be even. For example, if the WDATA field has the value 0000_0001h, the value of the parity field needs to be 1 so that the total number of 1s is even. It also reports errors to DCM in case of discrepancies.

36.2.1.2 Triple voting

DCF clients that use triple voting have three copies of the register. DCM writes to all the three registers in a single write cycle. During chip operation, the DCF client continuously monitors whether all these three copies match. In case of discrepancies, the client reports errors to DCM. The chip uses the majority result, so single errors do not affect the chip's operation. If case of single error, since the chip uses the majority result, there will be no impact to the chip's behavior.

36.2.2 DCF client modification rules

Depending on its role in the chip, a DCF client may implement one or a combination of modification rules. If a modification rule is in effect, the order in which DCF records are placed in the record list may be important.

36.2.2.1 Write once

A DCF client using the write once rule can only be written with a single DCF record. The records that are appended later in the list are ignored and do not change the value of the client.

36.2.2.2 Write 0 only

A field in a DCF client can only be changed from 1 to 0. Therefore, if the value of a field in the previous DCF record is 0, an attempt by a later record to write 1 to it is ignored.

36.2.2.3 Write 1 only

A field in a DCF client can only be changed from 0 to 1. Therefore, if the value of a field in the previous DCF record is 1, an attempt by a later record to write 0 to the field is ignored.

36.3 DCF record structure

A DCF record is a double-word (64-bit) entry that consists of the following:

- Control word—This provides information to locate the corresponding DCF client internal to the chip (pointer to the location of a register internal to the chip).
- Data word—This contains the data to be written to the DCF client.

DCF records select the target DCF client using a 30-bit field in the DCF record that consists of a 15-bit chip select field and a 15-bit address field. All modules that include DCF clients are assigned a chip select during chip definition. The address field is only relevant for address decoding within that module and may not necessarily relate to the address of a register that is visible to you.

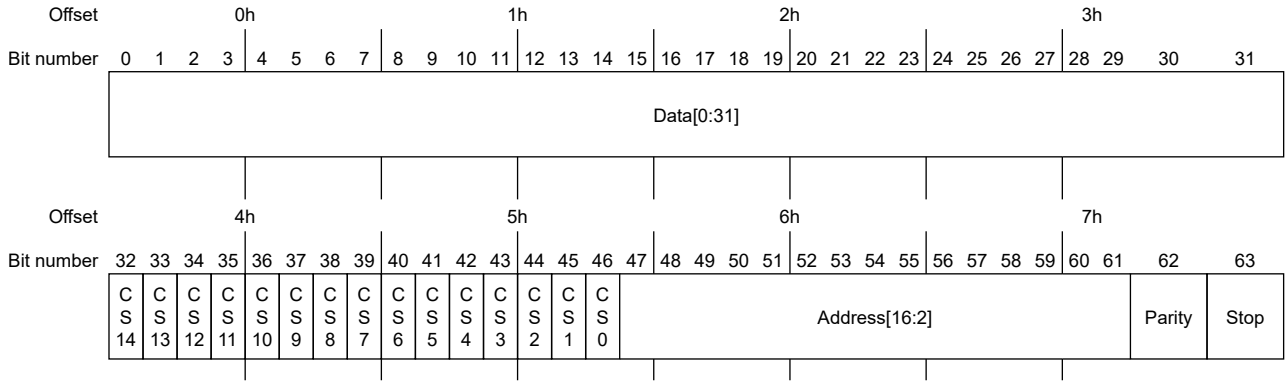


Figure 141. DCF record structure

Table 186. DCF record field descriptions

Field	Name	Description
0-31	Data[0:31]	Provides the data that is to be written to the DCF client.
32-46	CS*n*	Indicates chip select *n*. The value 1 is written to a chip select per DCF record to select the target module for the DCF client. The value 0 must be written to all other chip selects.

Table continues on the next page...

Table 186. DCF record field descriptions (continued)

Field	Name	Description
47-61	Address[16:2]	Contains the address of the DCF client within the selected module. Address decoding for DCF clients may not match the standard software address map decoding. For details, see the DCF client addresses provided with each module.
62	Parity	Indicates parity for the DCF record.
63	Stop	Indicates the end of the list for DCF records. 0 – Not the end 1 – End The erased state of flash memory is FFFF_FFFF_FFFF_FFFFh. Therefore, the list ends with the first unprogrammed double word. This location can be programmed with a new record to extend the list.

36.4 DCF records sequence

An individual DCF record contains information to locate the corresponding DCF client internal to the chip (control word) and the data to be written to that client (data word).

DCF records appear as contiguous series of entries programmed at a specific address within UTEST flash memory, and must present the following pattern:

- The first DCF record must be a start record. This record must be placed at the beginning of a DCF area in UTEST flash memory to indicate to the chip that the specified records must be processed.
- DCF records containing configuration data must immediately follow the start record with no blank records in between. An unprogrammed record is interpreted as a stop record and no DCF records following that are processed. This allows you to program the records in several sessions, appending new records at the end of the list each time.
- DCF stop record with bit set indicates the end of configuration records. It is not recommended to set STOP bit in last DCF record programmed during production because that prevents appending additional DCFs records. The UTEST flash memory location following the last DCF record programmed at the factory is an unprogrammed location, which has FFFF_FFFFh as its content. Thus, the stop bit location in this unprogrammed flash memory location is logic 1, signifying that this is the last DCF record and it is not to be acted upon.

Table 187 shows the record that DCM recognizes as a start record.

Table 187. DCF start record

0h (0:31)	4h (32:63)
05AA_55AFh	0000_0000h

The factory sets the DCF start record at the beginning of the UTEST flash memory area.

Table 188 shows the record that DCM recognizes as a stop record.

Table 188. DCF stop record

0:31	32:62	63
Ignored	Ignored	1

The DCF records that you supply must be added in a contiguous manner immediately following the factory-written DCF records. You must never have an unprogrammed record in the series of DCF records because that is interpreted as a stop record.

Table 189 shows the series of DCF records when *n* data records are stored in the UTEST flash memory.

Table 189. Series of DCF records in UTEST flash memory

Record type	Address offset	Data			
Start	0h	05AA_55AFh			
	4h	0000_0000h			STOP = 0
Data	8h	WDATA[31:0]			
	Ch	CS[14:0]	ADDR[16:2]	Parity	STOP = 0
	10h	WDATA[31:0]			
	14h	CS[14:0]	ADDR[16:2]	Parity	STOP = 0
	...	...			
Stop	8(*n*-1) + 0h	Reserved			
	8(*n*-1) + 4h	Reserved			1
Ignored	8*n* + 0h				
	8*n* + 4h				

More than one DCF records can write to the same DCF client. In this case, the later record usually overrides a DCF client value defined by a previous record. However, not all DCF clients allow overwrites; this depends on individual implementation of DCF clients.

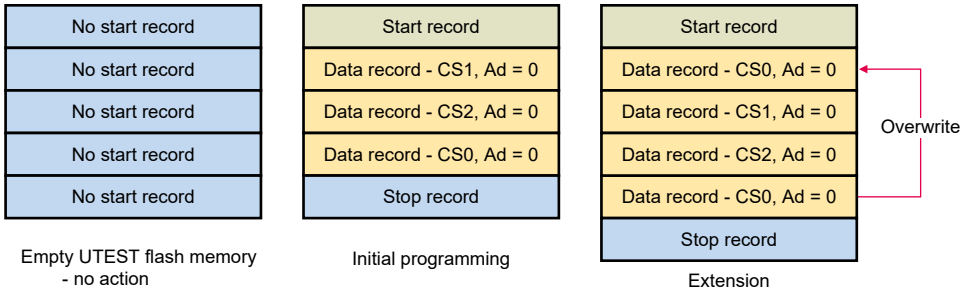


Figure 142. Appending DCF records

36.5 Chip configuration records

The DCF clients table contains information on DCF clients available in the chip. See the DCF clients file attached to this document.

The next table shows an example of how the information in this chapter is integrated with the attached S32K396_DCF_clients file.

Table 190. Integration of DCF information

Type	Data(n) assuming Quad page program	Data	Comment
Reset pad dedicated control DCF client (dcf_client_reset_pad_dedicated - column D in the "Utest DCF Clients" sheet)	Data word (determined base on the DCF record description in the "Utest DCF Client Register Bits" sheet)	0000_0001h	Data to enable pad as dedicated reset pad
	Control word (without parity) (selected from column C in the "Utest DCF Clients" sheet)	0010_0008h	Chip Select is 3 and address is 8, 0010_0000h+ 8h

36.6 Glossary

UTEST User test. Refers to UTEST region of the flash.

Chapter 37
Device Configuration Module General-Purpose Registers (DCM_GPR)

37.1 DCM controlled features and availability in product family

Based on the chip features described in ‘Feature comparison’ section in ‘Introduction’ chapter, there are some features which are present only in specific parts in the MWCT2xxxS product family. The following table summarizes the corresponding DCM register fields along with the parts wherein the corresponding register fields are available. In rest of the parts within the product family, the corresponding fields are reserved.

Table 191. DCM controlled features and availability in product family

Register field	Register field abbreviation	Parts wherein this field is available
DCMRWD3[1]	CM7_1_LOCKUP_EN	MWCT2D17S, MWCT2D16S
DCMRWD3[5]	TCM_GSKT_ALARM_EN	MWCT2D17S, MWCT2D16S, MWCT2016S, MWCT2015S
DCMRWD3[6]	DMA_SYS_GSKT_ALARM_EN	MWCT2D17S, MWCT2D16S
DCMRWD3[7]	DMA_PERIPH_GSKT_ALARM_EN	MWCT2D17S, MWCT2D16S
DCMRWD3[9]	DMA_AXBS_ALARM_EN	MWCT2D17S, MWCT2D16S
DCMRWD3[12]	QSPI_GSKT_ALARM_EN	MWCT2D17S, MWCT2D16S
DCMRWD3[14]	AIPS2_GSKT_ALARM_EN	MWCT2D17S, MWCT2D16S
DCMRWD3[17]	TCM_AXBS_ALARM_EN	MWCT2D17S, MWCT2D16S
DCMRWD3[18]	EMAC_GSKT_ALARM_EN	MWCT2D17S, MWCT2D16S
DCMRWD3[19]	PERIPH_AXBS_ALARM_EN	MWCT2D17S, MWCT2D16S
DCMRWD3[24]	PRAM1_ECC_ERR_EN	MWCT2D17S
DCMRWD3[27]	CM7_1_DCDATA_ECC_ERR_EN	MWCT2D17S, MWCT2D16S
DCMRWD3[29]	CM7_1_DCTAG_ECC_ERR_EN	MWCT2D17S, MWCT2D16S
DCMRWD3[31]	CM7_1_ICDATA_ECC_ERR_EN	MWCT2D17S, MWCT2D16S
DCMRWD4[1]	CM7_1_ICTAG_ECC_ERR_EN	MWCT2D17S, MWCT2D16S
DCMRWD4[5]	CM7_1_ITCM_ECC_ERR_EN	MWCT2D17S, MWCT2D16S
DCMRWD4[6]	CM7_1_DTCM0_ECC_ERR_EN	MWCT2D17S, MWCT2D16S
DCMRWD4[7]	CM7_1_DTCM1_ECC_ERR_EN	MWCT2D17S, MWCT2D16S
DCMRWD4[10]	PRAM1_FCCU_ALARM_EN	MWCT2D17S
DCMRWD4[16]	PF2_CODE_ECC_ERR_EN	MWCT2D17S, MWCT2D16S
DCMRWD4[17]	PF2_DATA_ECC_ERR_EN	MWCT2D17S, MWCT2D16S
DCMRWD5[14]	TCM_RDATA_EDC_ERR_EN	MWCT2D17S, MWCT2D16S
DCMRWD5[15]	EMAC_RDATA_EDC_ERR_EN	MWCT2D17S, MWCT2D16S

Table continues on the next page...

Table 191. DCM controlled features and availability in product family (continued)

DCMRWD5[18]	CM7_1_AHBP_RDATA_EDC_ERR_EN	MWCT2D17S, MWCT2D16S
DCMRWD5[19]	CM7_1_AHBM_RDATA_EDC_ERR_EN	MWCT2D17S, MWCT2D16S
DCMRWD6[6]	eMIOS2_DBG_DIS_CM7_0	MWCT2D17S
DCMRWD6[9]	SWT1_DBG_DIS_CM7_0	MWCT2D17S, MWCT2D16S
DCMRWD6[11]	STM1_DBG_DIS_CM7_0	MWCT2D17S, MWCT2D16S
DCMRWD6[14]	PIT2_DBG_DIS_CM7_0	MWCT2D17S, MWCT2D16S
DCMRWD6[19]	LPSPI4_DBG_DIS_CM7_0	MWCT2D17S
DCMRWD6[20]	LPSPI5_DBG_DIS_CM7_0	MWCT2D17S
DCMRWD6[27]	FLEXCAN3_DBG_DIS_CM7_0	MWCT2D17S, MWCT2D16S, MWCT2016S
DCMRWD6[28]	FLEXCAN4_DBG_DIS_CM7_0	MWCT2D17S, MWCT2016S
DCMRWD6[29]	FLEXCAN5_DBG_DIS_CM7_0	MWCT2D17S, MWCT2016S
DCMRWD6[30]	SAI0_DBG_DIS_CM7_0	MWCT2D17S, MWCT2D16S
DCMRWD6[31]	SAI1_DBG_DIS_CM7_0	MWCT2D17S, MWCT2D16S
DCMRWD8[0]	EDMA_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[1]	FCCU_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[2]	LCU0_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[3]	LCU1_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[4]	eMIOS0_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[5]	eMIOS1_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[6]	eMIOS2_DBG_DIS_CM7_1	MWCT2D17S
DCMRWD8[7]	RTC_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[8]	SWT0_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[9]	SWT1_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[10]	STM0_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[11]	STM1_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[12]	PIT0_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[13]	PIT1_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[14]	PIT2_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[15]	LPSPI0_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[16]	LPSPI1_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[17]	LPSPI2_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[18]	LPSPI3_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[19]	LPSPI4_DBG_DIS_CM7_1	MWCT2D17S

Table continues on the next page...

Table 191. DCM controlled features and availability in product family (continued)

DCMRWD8[20]	LPSPI5_DBG_DIS_CM7_1	MWCT2D17S
DCMRWD8[21]	LPI2C0_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[22]	LPI2C1_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[23]	FLEXIO_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[24]	FLEXCAN0_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[25]	FLEXCAN1_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[26]	FLEXCAN2_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[27]	FLEXCAN3_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[28]	FLEXCAN4_DBG_DIS_CM7_1	MWCT2D17S
DCMRWD8[29]	FLEXCAN5_DBG_DIS_CM7_1	MWCT2D17S
DCMRWD8[30]	SAI0_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWD8[31]	SAI1_DBG_DIS_CM7_1	MWCT2D17S, MWCT2D16S
DCMRWF1[0]	CAN_TIMESTAMP_SEL	MWCT2D17S, MWCT2D16S
DCMRWF1[1]	CAN_TIMESTAMP_EN	MWCT2D17S, MWCT2D16S
DCMRWF1[7]	EMAC_CONF_SEL	MWCT2D17S, MWCT2D16S
DCMRWF1[15]	VDD_HV_B_IO_CTRL_LATCH	MWCT2D17S, MWCT2D16S
DCMRWF1[26]	VDD_HV_B_VLT_DVDR_EN	MWCT2D17S, MWCT2D16S
DCMRWF1[27]	VDD_1_5_VLT_DVDR_EN	MWCT2D17S, MWCT2D16S
DCMRWF1[31]	EMAC_TX_RMII_CLK_LPBCK_EN	MWCT2D17S, MWCT2D16S
DCMRWF4[0]	MUX_MODE_EN_ADC1_S18	MWCT2015S
DCMRWF4[5]	MUX_MODE_EN_ADC1_S22	MWCT2D17S, MWCT2D16S, MWCT2016S
DCMRWF4[6]	MUX_MODE_EN_ADC1_S23	MWCT2D17S, MWCT2D16S, MWCT2016S
DCMRWF4[7]	MUX_MODE_EN_ADC0_S12	MWCT2015S
DCMRWF4[8]	MUX_MODE_EN_ADC0_S13	MWCT2015S
DCMRWF4[9]	MUX_MODE_EN_ADC2_S8	MWCT2D17S
DCMRWF4[10]	MUX_MODE_EN_ADC2_S9	MWCT2D17S
DCMRWF4[11]	MUX_MODE_EN_ADC0_S14	MWCT2015S
DCMRWF4[12]	MUX_MODE_EN_ADC0_S17	MWCT2015S
DCMRWF4[18]	CM7_1_CPUWAIT	MWCT2D17S, MWCT2D16S
DCMRWF4[28]	MUX_MODE_EN_ADC0_P2	MWCT2015S
DCMROD3[1]	CM7_1_LOCKUP	MWCT2D17S, MWCT2D16S

Table continues on the next page...

Table 191. DCM controlled features and availability in product family (continued)

DCMROD3[5]	TCM_GSKT_ALARM	MWCT2D17S, MWCT2D16S, MWCT2016S, MWCT2015S
DCMROD3[6]	DMA_SYS_GSKT_ALARM	MWCT2D17S, MWCT2D16S
DCMROD3[7]	DMA_PERIPH_GSKT_ALARM	MWCT2D17S, MWCT2D16S
DCMROD3[9]	DMA_AXBS_ALARM	MWCT2D17S, MWCT2D16S
DCMROD3[12]	QSPI_GSKT_ALARM	MWCT2D17S, MWCT2D16S
DCMROD3[14]	AIPS2_GSKT_ALARM	MWCT2D17S, MWCT2D16S
DCMROD3[17]	TCM_AXBS_ALARM	MWCT2D17S, MWCT2D16S,
DCMROD3[18]	EMAC_GSKT_ALARM	MWCT2D17S, MWCT2D16S
DCMROD3[19]	PERIPH_AXBS_ALARM	MWCT2D17S, MWCT2D16S
DCMROD3[24]	PRAM1_ECC_ERR	MWCT2D17S
DCMROD3[27]	CM7_1_DCDATA_ECC_ERR	MWCT2D17S, MWCT2D16S
DCMROD3[29]	CM7_1_DCTAG_ECC_ERR	MWCT2D17S, MWCT2D16S
DCMROD3[31]	CM7_1_ICDATA_ECC_ERR	MWCT2D17S, MWCT2D16S
DCMROD4[1]	CM7_1_ICTAG_ECC_ERR	MWCT2D17S, MWCT2D16S
DCMROD4[5]	CM7_1_ITCM_ECC_ERR	MWCT2D17S, MWCT2D16S
DCMROD4[6]	CM7_1_DTCM0_ECC_ERR	MWCT2D17S, MWCT2D16S
DCMROD4[7]	CM7_1_DTCM1_ECC_ERR	MWCT2D17S, MWCT2D16S
DCMROD4[10]	PRAM1_FCCU_ALARM	MWCT2D17S
DCMROD4[16]	PF2_CODE_ECC_ERR	MWCT2D17S, MWCT2D16S
DCMROD4[17]	PF2_DATA_ECC_ERR	MWCT2D17S, MWCT2D16S
DCMROD5[14]	TCM_RDATA_EDC_ERR	MWCT2D17S, MWCT2D16S
DCMROD5[15]	EMAC_RDATA_EDC_ERR	MWCT2D17S, MWCT2D16S
DCMROD5[18]	CM7_1_AHBP_RDATA_EDC_ERR	MWCT2D17S, MWCT2D16S
DCMROD5[19]	CM7_1_AHBM_RDATA_EDC_ERR	MWCT2D17S, MWCT2D16S
DCMROF1[0]	EMAC_MDC_CHID_0	MWCT2D17S, MWCT2D16S
DCMROF1[1]	EMAC_MDC_CHID_1	MWCT2D17S, MWCT2D16S
DCMROF20[1]	LMAUTO_DIS	MWCT2D17S, MWCT2D16S
DCMROF20[3]	DMA_AXBS_IAHB_BYP	MWCT2D17S, MWCT2D16S
DCMROF20[5]	QSPI_IAHB_BYP	MWCT2D17S, MWCT2D16S

37.2 DCM_GPR register descriptions

Before you start to work with the GPR register take care about the following:

- Do not modify the reserved locations, registers, or reserved bits with respect to their default configurations. Chip behavior is not guaranteed in case of such writes.

- The writes to the DCM Read Write registers (DCMRWPx, DCMRWDx, DCMRWFx and DCMRWPPx) are synchronized and take up to 4 cycles of CORE_CLK. Hence the user should wait for at least 4 CORE_CLK cycles after configuring these registers for these to get effective.
- The DCM registers are present in Standby domain and retain values across Standby mode. Therefore, reading the status registers after exiting Standby mode may indicate a previously latched value. To read these status registers after a reset event or after exiting Standby mode, you must first write 0 to the fields of these registers. This clears the previous status or transient information because of the reset event, and specifies the correct status.
- You must access DCM after at least 92 AIPS_SLOW_CLK cycles of writing to MC_RGM.ERCTRL because the configuration of MC_RGM.ERCTRL takes several cycles to be effective. Also, the DCMRW *n* registers are synchronized; therefore, changes to them may take up to four CORE_CLK cycles to be effective.

37.2.1 DCM_GPR memory map

DCM_GPR base address: 402A_C000h

Offset	Register	Width (In bits)	Access	Reset value
200h	Read-Only GPR On Destructive Reset 1 (DCMROD1)	32	RW	0000_0000h
208h	Read-Only GPR On Destructive Reset 3 (DCMROD3)	32	RW	0000_0000h
20Ch	Read-Only GPR On Destructive Reset 4 (DCMROD4)	32	RW	0000_0000h
210h	Read-Only GPR On Destructive Reset 5 (DCMROD5)	32	RW	0000_0000h
300h	Read-Only GPR On Functional Reset 1 (DCMROF1)	32	RW	0000_0000h
304h	Read-Only GPR On Functional Reset 2 (DCMROF2)	32	RW	0000_0000h
308h	Read-Only GPR On Functional Reset 3 (DCMROF3)	32	RW	0000_0000h
30Ch	Read-Only GPR On Functional Reset 4 (DCMROF4)	32	RW	0000_0000h
310h	Read-Only GPR On Functional Reset 5 (DCMROF5)	32	RW	0000_0000h
314h	Read-Only GPR On Functional Reset 6 (DCMROF6)	32	RW	0000_0000h
318h	Read-Only GPR On Functional Reset 7 (DCMROF7)	32	RW	0000_0000h
31Ch	Read-Only GPR On Functional Reset 8 (DCMROF8)	32	RW	0000_0000h
320h	Read-Only GPR On Functional Reset 9 (DCMROF9)	32	RW	0000_0000h
324h	Read-Only GPR On Functional Reset 10 (DCMROF10)	32	RW	0000_0000h
328h	Read-Only GPR On Functional Reset 11 (DCMROF11)	32	RW	0000_0000h
32Ch	Read-Only GPR On Functional Reset 12 (DCMROF12)	32	RW	0000_0000h
330h	Read-Only GPR On Functional Reset 13 (DCMROF13)	32	RW	0000_0000h
334h	Read-Only GPR On Functional Reset 14 (DCMROF14)	32	RW	0000_0000h
338h	Read-Only GPR On Functional Reset 15 (DCMROF15)	32	RW	0000_0000h
33Ch	Read-Only GPR On Functional Reset 16 (DCMROF16)	32	RW	0000_0000h
340h	Read-Only GPR On Functional Reset 17 (DCMROF17)	32	RW	0000_0000h
348h	Read-Only GPR On Functional Reset 19 (DCMROF19)	32	R	4000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
34Ch	Read-Only GPR On Functional Reset 20 (DCMROF20)	32	R	See section
350h	Read-Only GPR On Functional Reset 21 (DCMROF21)	32	R	0000_0000h
400h	Read Write GPR On POR 1 (DCMRWP1)	32	RW	0000_0400h
408h	Read Write GPR On POR 3 (DCMRWP3)	32	RW	0000_0000h
504h	Read Write GPR On Destructive Reset 2 (DCMRWD2)	32	RW	0000_0000h
508h	Read Write GPR On Destructive Reset 3 (DCMRWD3)	32	RW	FF4F_FBFFh
50Ch	Read Write GPR On Destructive Reset 4 (DCMRWD4)	32	RW	EF7F_FFFFh
510h	Read Write GPR On Destructive Reset 5 (DCMRWD5)	32	RW	007F_FFFFh
514h	Read Write GPR On Destructive Reset 6 (DCMRWD6)	32	RW	0000_0000h
518h	Read Write GPR On Destructive Reset 7 (DCMRWD7)	32	RW	0000_0000h
51Ch	Read Write GPR On Destructive Reset 8 (DCMRWD8)	32	RW	0000_0000h
520h	Read Write GPR On Destructive Reset 9 (DCMRWD9)	32	RW	0000_0000h
600h	Read Write GPR On Functional Reset 1 (DCMRWF1)	32	RW	0000_0000h
604h	Read Write GPR On Functional Reset 2 (DCMRWF2)	32	RW	0000_0000h
60Ch	Read Write GPR On Functional Reset 4 (DCMRWF4)	32	RW	0000_0000h
610h	Read Write GPR On Functional Reset 5 (DCMRWF5)	32	RW	See section
700h	Read-Only GPR On PMCPOR Reset 1 (DCMROPP1)	32	RW	0000_0000h
704h	Read-Only GPR On PMCPOR Reset 2 (DCMROPP2)	32	RW	0000_0000h
708h	Read-Only GPR On PMCPOR Reset 3 (DCMROPP3)	32	RW	0000_0000h
70Ch	Read-Only GPR On PMCPOR Reset 4 (DCMROPP4)	32	RW	0000_0000h

37.2.2 Read-Only GPR On Destructive Reset 1 (DCMROD1)

Offset

Register	Offset
DCMROD1	200h

Function

Contains information related to:

- Key response ready status.
- DCF violation from HSE_B.
- PCU input isolation status.

This register resets after destructive reset 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved													Reserved	Reserved	PCU_ISO...
W	W1C													W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-3 —	Reserved
2 —	Reserved
1 —	Reserved
0 PCU_ISO_STATUS	PCU Input Isolation Status On Previous Standby Entry Specifies whether input isolation was enabled in the previous standby entry. 0b - No 1b - Yes

37.2.3 Read-Only GPR On Destructive Reset 3 (DCMROD3)

Offset

Register	Offset
DCMROD3	208h

Function

Contains information related to:

- ECC and life cycle errors.
- AXBS, RCCU, and gasket alarms.

This register resets after destructive reset 3.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CM7_1_I...	CM7_0_I...	CM7_1_D...	CM7_0_D...	CM7_1_D...	CM7_0_D...	PRAM0_E...	PRAM1_E...	Reserved	LC_ERR	Reserved		PERIPH_...	EMAC_GS...	TCM_AXB...	DATA_ED...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C		W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR_ED...	AIPS2_G...	AIPS1_G...	QSPI_GS...	HSE_GSK...	Reserved	DMA_AXB...	SYS_AXB...	DMA_PER...	DMA_SYS...	TCM_GSK...	Reserved	Reserved	HSE_LOC...	CM7_1_L...	CM7_0_L...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 CM7_1_ICDATA_ECC_ERR	Cortex-M7_1 I-cache Multi-Bit ECC Error Specifies whether the Cortex-M7_1 core's I-cache data memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU noncritical fault (NCF) 2. 0b - No 1b - Yes
30 CM7_0_ICDATA_ECC_ERR	Cortex-M7_0 I-cache Data ECC Error Specifies whether the Cortex-M7_0 core's I-cache data memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
29 CM7_1_DCTAG_ECC_ERR	Cortex-M7_1 D-cache Tag ECC Error Specifies whether the Cortex-M7_1 core's D-cache tag memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
28 CM7_0_DCTAG_ECC_ERR	Cortex-M7_0 D-cache Tag ECC Error Specifies whether the Cortex-M7_0 core's D-cache tag memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 CM7_1_DCDATA_ECC_ERR	Cortex-M7_1 D-cache Data Memory ECC Error Specifies whether the Cortex-M7_1 core's D-cache data memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
26 CM7_0_DCDATA_ECC_ERR	Cortex-M7_0 D-cache Data Memory ECC Error Specifies whether the Cortex-M7_0 core's D-cache data memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
25 PRAM0_ECC_ERR	Multi-Bit ECC Error From PRAM0 Specifies whether a multi-bit ECC error occurred from PRAM0. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
24 PRAM1_ECC_ERR	Multi-Bit ECC Error From PRAM1 Specifies whether a multi-bit ECC error occurred from PRAM1. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
23 —	Reserved
22 LC_ERR	Error In Life Cycle Scanning Specifies whether an error occurred during life-cycle scanning. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No error while lifecycle scanning. 1b - Error while lifecycle scanning
21-20 —	Reserved
19	Peripheral AXBS_Lite Safety Alarm Status Specifies whether peripheral AXBS_Lite reported a safety alarm.

Table continues on the next page...

Table continued from the previous page...

Field	Function
PERIPH_AXBS_ALARM	Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
18 EMAC_GSKT_ALARM	EMAC IAHB Gasket Alarm Status Specifies whether the EMAC IAHB gasket reported an alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
17 TCM_AXBS_ALARM	TCM AHB Splitter Safety Alarm Status Specifies whether the TCM AHB splitter reported a safety alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
16 DATA_EDC_ERR	Data EDC Error Specifies whether an integrity error occurred on address for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
15 ADDR_EDC_ERR	Address EDC Error Status Specifies whether an integrity error occurred on address for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
14 AIPS2_GSKT_ALARM	AIPS2 IAHB Gasket Alarm Status. Read this bit to identify the reason for a fault in case of FCCU NCF 1. 0b - No alarm indicated by AIPS2 IAHB gasket. 1b - Alarm indicated by AIPS2 IAHB gasket.
13 AIPS1_GSKT_ALARM	AIPS1 IAHB Gasket Alarm Status. Read this bit to identify the reason for a fault in case of FCCU NCF 1. 0b - No alarm indicated by AIPS1 IAHB gasket. 1b - Alarm indicated by AIPS1 IAHB gasket.
12 QSPI_GSKT_ALARM	QuadSPI IAHB Gasket Alarm Status Specifies whether the QuadSPI IAHB gasket reported an alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No 1b - Yes
11 HSE_GSKT_ALARM	HSE IAHB Gasket Alarm Status Specifies whether the HSE_B IAHB gasket reported an alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
10 —	Reserved
9 DMA_AXBS_ALARM	eDMA AXBS_Lite Safety Alarm Status Specifies whether eDMA AXBS_Lite reported a safety alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
8 SYS_AXBS_ALARM	System AXBS Safety Alarm Status Specifies whether the system AXBS indicated a safety alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
7 DMA_PERIPH_GSKT_ALARM	eDMA Peripheral Gasket Alarm Status Specifies whether the eDMA peripheral AXBS IAHB gasket reported a safety alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
6 DMA_SYS_GSKT_ALARM	eDMA System Gasket Alarm Status Specifies whether the IAHB gasket safety alarm, from the eDMA system AXBS IAHB gasket, reported a safety alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
5	TCM IAHB Gasket Monitor Alarm Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
TCM_GSKT_ALARM	Specifies whether the TCM IAHB gasket reported an alarm. If the value of this field is 1, the gasket reports a monitor alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
4 —	Reserved
3 —	Reserved
2 HSE_LOCKUP	HSE_B Core Lockup Status Specifies whether the HSE_B core is in the Lockup state. Read this field to identify the reason for a fault in case of FCCU NCF 0. 0b - No 1b - Yes
1 CM7_1_LOCKUP	Cortex-M7_1 Core Lockup Status Specifies whether the Cortex_M7_1 core is in the Lockup state. Read this field to identify the reason for a fault in case of FCCU NCF 0. 0b - No 1b - Yes
0 CM7_0_LOCKUP	Cortex-M7_0 Core Lockup Status Specifies whether the Cortex-M7_0 core is in the Lockup state. Read this field to identify the reason for a fault in case of FCCU NCF 0. 0b - No 1b - Yes

37.2.4 Read-Only GPR On Destructive Reset 4 (DCMROD4)

Offset

Register	Offset
DCMROD4	20Ch

Function

Contains information related to:

- Accidental partial test activation errors.
- Go/No-go indicator supply statuses.
- Flash memory errors.
- Alarm statuses.

This register resets after destructive reset 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	TEST_AC...	TEST_AC...	Reserved	VDD2P5_...	VDD1P1_...	FLASH_E...	Reserved	Reserved	FLASH_S...	FLASH_R...	FLASH_R...	FLASH_A...	FLASH_E...	PF2_DAT...	PF2_COD...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PF1_DAT...	PF1_COD...	PF0_DAT...	PF0_COD...	HSE_RAM...	PRAM1_F...	PRAM0_F...	DMA_TCD...	CM7_1_D...	CM7_1_D...	CM7_1_I...	CM7_0_D...	CM7_0_D...	CM7_0_I...	CM7_1_I...	CM7_0_I...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 TEST_ACTIVATION_1_ERR	Accidental Partial Test Activation 1 Error Specifies whether partial test 1 is activated accidentally. Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No 1b - Yes
29 TEST_ACTIVATION_0_ERR	Accidental Partial Test Activation 0 Error Specifies whether partial test 0 is activated accidentally. Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No 1b - Yes
28 —	Reserved
27	Go/No-go Indicator For VDD_HV_FLA Specifies whether the VDD_HV_FLA (double-bond) supply going to XOSC and PLL is clean.

Table continues on the next page...

Table continued from the previous page...

Field	Function
VDD2P5_GNG_ERR	Read this field to identify the reason for a fault in case of FCCU NCF 4. If this field = 1, the "go" indication specifies a clean supply, and if this field = 0, the "no-go" indication specifies an unclean supply with a fault in the double-bond connection or its routing within the chip. 0b - Yes 1b - No
26 VDD1P1_GNG_ERR	Go/No-go Indicator For VDD1PD1 Specifies whether the VDD1PD1 (double-bond) supply going to PLL is clean. Read this field to identify the reason for a fault in case of FCCU NCF 4. If this field = 1, the "go" indication specifies a clean supply, and if this field = 0, the "no-go" indication specifies an unclean supply with a fault in the double-bond connection or its routing within the chip. 0b - Yes 1b - No
25 FLASH_ECC_ERR	ECC Error From Flash Controller This alarm Specifies that the flash controller detected an error in the address ECC manipulation logic through EDC. Read this bit to identify the reason for a fault in case of FCCU NCF 3. 0b - No ECC error from flash controller. 1b - ECC error from flash controller.
24 —	Reserved
23 —	Reserved
22 FLASH_SCAN_ERR	Flash Memory Scan Error Status Specifies whether the flash memory encountered an error during the DCM flash scanning process because of invalid data. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
21 FLASH_RST_ERR	Flash Reset Error Status Specifies whether the flash memory encountered a flash memory reset error during its reset reads. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
20	Flash Memory Reference Error Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLASH_REF_ERR	Specifies whether the flash memory encountered a reference current loss or read voltage error during previous read(s). Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
19 FLASH_ADDR_ENC_ERR	Flash Memory Address Encode Error Status Specifies whether FMU reported an address encode error in the flash memory. During address decoding, if multiple or no address line is selected, FMU reports an address encode error. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
18 FLASH_EDC_ERR	Flash Memory EDC Error Status Specifies whether FMU reported an integrity error after an ECC correction error in the flash memory. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
17 PF2_DATA_ECC_ERR	Program Flash Memory 2 Data ECC Error Status Specifies whether FMU reported uncorrectable errors in the flash memory controller port 2 data memory. These errors are connected to FCCU NCFs and to ERM. See the "Error Reporting Module (ERM)" chapter for memory errors and mapping onto ERM channels. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
16 PF2_CODE_ECC_ERR	Program Flash Memory 2 Code ECC Error Status Specifies whether FMU reported uncorrectable errors in the flash memory controller port 2 code memory. These errors are connected to FCCU NCFs and to ERM. See the "Error Reporting Module (ERM)" chapter for memory errors and mapping onto ERM channels. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
15 PF1_DATA_ECC_ERR	Program Flash Memory 1 Data ECC Error Status Specifies whether FMU reported uncorrectable errors in the flash memory controller port 1 data memory. These errors are connected to FCCU NCFs and to ERM. See the "Error Reporting Module (ERM)" chapter for memory errors and mapping onto ERM channels.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
14 PF1_CODE_ECC_ERR	Program Flash Memory 1 Code ECC Error Status Specifies whether FMU reported uncorrectable errors in the flash memory controller port 1 code memory. These errors are connected to FCCU NCFs and to ERM. See the "Error Reporting Module (ERM)" chapter for memory errors and mapping onto ERM channels. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
13 PF0_DATA_ECC_ERR	Program Flash Memory 0 Data ECC Error Status Specifies whether FMU reported uncorrectable errors in the flash memory controller port 0 data memory. These errors are connected to FCCU NCFs and to ERM. See the "Error Reporting Module (ERM)" chapter for memory errors and mapping onto ERM channels. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
12 PF0_CODE_ECC_ERR	Program Flash Memory 0 Code ECC Error Status Specifies whether FMU reported uncorrectable errors in the flash memory controller port 0 code memory. These errors are connected to FCCU NCFs and to ERM. See the "Error Reporting Module (ERM)" chapter for memory errors and mapping onto ERM channels. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
11 HSE_RAM_ECC_ERR	HSE_B RAM Uncorrectable ECC Status Specifies whether HSE_B RAM reported an uncorrectable ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
10 PRAM1_FCCU_ALARM	PRAM1 FCCU Alarm Status Specifies whether PRAM1 reported a safety alarm. This field specifies the status of the PRAM1 safety alarm, whether the alarm is set on faulty PRAM1 read or read-modify error. Read this field to identify the reason for a fault in case of FCCU NCF 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No 1b - Yes
9 PRAM0_FCCU_ALARM	PRAM0 FCCU Alarm Status Specifies whether PRAM0 reported a safety alarm. This field specifies the status of the PRAM0 safety alarm, whether the alarm is set on faulty PRAM0 read or read-modify error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
8 DMA_TCD_RAM_ECC_ERR	eDMA TCD RAM ECC Error Specifies whether the eDMA_TCD memory detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
7 CM7_1_DTCM1_ECC_ERR	Cortex-M7_1 DTCM 1 ECC Error Specifies whether the Cortex-M7_1 core's DTCM block 1 detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_1 core's DTCM consists of two physical blocks. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
6 CM7_1_DTCM0_ECC_ERR	Cortex-M7_1 DTCM 0 ECC Error Specifies whether the Cortex-M7_1 core's DTCM block 0 detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_1 core's DTCM consists of two physical blocks. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
5 CM7_1_ITCM_ECC_ERR	Cortex-M7_1 ITCM ECC Error Specifies whether the Cortex-M7_1 core's ITCM detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 CM7_0_DTCM1_ECC_ERR	Cortex-M7_0 DTCM 1 ECC Error Specifies whether the Cortex-M7_0 core's DTCM block 1 detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_0 core's DTCM consists of two physical blocks. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
3 CM7_0_DTCM0_ECC_ERR	Cortex-M7_0 DTCM 0 ECC Error Specifies whether the Cortex-M7_0 core's DTCM block 0 detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_0 core's DTCM consists of two physical blocks. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
2 CM7_0_ITCM_ECC_ERR	Cortex-M7_0 ITCM ECC Error Specifies whether the Cortex-M7_0 core's ITCM detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. Read this field to identify the reason of fault in case of FCCU NCF 2. 0b - No 1b - Yes
1 CM7_1_ICTAG_ECC_ERR	Cortex-M7_1 I-cache Tag ECC Error Specifies whether the Cortex-M7_1 core's I-cache tag memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
0 CM7_0_ICTAG_ECC_ERR	Cortex-M7_0 I-cache Tag ECC Error Specifies whether the Cortex-M7_0 core's I-cache tag memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes

37.2.5 Read-Only GPR On Destructive Reset 5 (DCMROD5)

Offset

Register	Offset
DCMROD5	210h

Function

Contains information related to:

- ECC and EDC errors.
- Activation and bus errors.

This register resets after destructive reset 5.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	HSE_RDA...	CM7_0_A...	CM7_0_A...	CM7_1_A...	CM7_1_A...	DMA_RDA...	Reserved
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EMAC_RD...	TCM_RDA...	DEBUG_A...	MCT_BUS...	STCU_BI...	MBIST_A...	STCU_NCF	SW_NCF_3	SW_NCF_2	SW_NCF_1	SW_NCF_0	INTM_3_...	INTM_2_...	INTM_1_...	INTM_0_...	TEST_AC...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 HSE_RDATA_EDC_ERR	HSE_B Read Data EDC Error Specifies whether an integrity error is reported on the HSE_B read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
21 CM7_0_AHBM_RDATA_EDC_ERR	Cortex-M7_0 AHBM Read Data EDC Error Specifies whether an integrity error is reported on the Cortex-M7_0 core's main read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
20 CM7_0_AHBP_RDATA_EDC_ERR	Cortex-M7_0 AHBP Read Data EDC Error Specifies whether an integrity error is reported on the Cortex-M7_0 core's peripheral read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
19 CM7_1_AHBM_RDATA_EDC_ERR	Cortex-M7_1 AHBM Read Data EDC Error Specifies whether an integrity error is reported on the Cortex-M7_1 core's main read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
18	Cortex-M7_1 AHBP Read Data EDC Error Specifies whether an integrity error is reported on the Cortex-M7_1 core's peripheral read data for safety.

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_1_AHBP_RDATA_EDC_ERR	Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
17 DMA_RDATA_EDC_ERR	eDMA Read Data EDC Error Specifies whether an integrity error is reported on the eDMA read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
16 —	Reserved
15 EMAC_RDATA_EDC_ERR	EMAC Read Data EDC Error Specifies whether an integrity error is reported on the EMAC read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
14 TCM_RDATA_EDC_ERR	TCM Read Data EDC Error Specifies whether an integrity error is reported on the TCM read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
13 DEBUG_ACTIVATION_ERR	Debug Activation Error Specifies whether an unintended debug is activated. This field monitors unintended debug activation. It displays 1 as its value when the core is in halted state with the application debug or debugger request not enabled. Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No 1b - Yes
12 MCT_BUS_ERR	MCT Bus Error Fault reported due to illegal access on MBIST controller (MCT). This fault is reported via a transfer error indication to the system. Read this bit to identify the reason of fault in case of FCCU NCF 5. 0b - No transfer error indicated from MCT. 1b - Transfer error indicated from MCT.

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 STCU_BIST_USER_CF	STCU2 BIST User Critical Fault (CF) Specifies whether MBIST is enabled accidentally when the fault condition is detected in Run mode. Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No 1b - Yes
10 MBIST_ACTIVATION_ERR	MBIST Activation Error Specifies whether an accidental backdoor access is enabled on memories. DCMRWD5[MBIST_ACTIVATION_ERR_EN] needs to be disabled on FCCU when performing a fault injection. Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No 1b - Yes
9 STCU_NCF	STCU2 NCF Result Error Specifies whether STCU2 NCF, which is a BIST result error, is reported. Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No 1b - Yes
8 SW_NCF_3	Software NCF3 Status Specifies whether DCMRWF1[FCCU_SW_NCF3] is enabled. Read this field to identify the reason for a fault in case of FCCU NCF 7. 0b - No 1b - Yes
7 SW_NCF_2	Software NCF2 Status Specifies whether DCMRWF1[FCCU_SW_NCF2] is enabled. Read this field to identify the reason for a fault in case of FCCU NCF 7. 0b - No 1b - Yes
6 SW_NCF_1	Software NCF1 Status Specifies whether DCMRWF1[FCCU_SW_NCF1] is enabled. Read this field to identify the reason for a fault in case of FCCU NCF 7. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 SW_NCF_0	Software NCF 0 Status Specifies whether DCMRWF1[FCCU_SW_NCF0] is enabled. Read this field to identify the reason for a fault in case of FCCU NCF 7. 0b - No 1b - Yes
4 INTM_3_ERR	INTM_3 Error Specifies whether INTM_3 reported an error. The reported error is recorded in INTM.INTM_STATUS3. See the "Functional description" section of the "Interrupt Monitor (INTM)" chapter for details. Read this field to identify the reason for a fault in case of FCCU NCF 6. 0b - No 1b - Yes
3 INTM_2_ERR	INTM_2 Error Specifies whether INTM_2 reported an error. The reported error is recorded in INTM.INTM_STATUS2. See the "Functional description" section of the "Interrupt Monitor (INTM)" chapter for details. Read this field to identify the reason for a fault in case of FCCU NCF 6. 0b - No 1b - Yes
2 INTM_1_ERR	INTM_1 Error Specifies whether INTM_1 reported an error. The reported error is recorded in INTM.INTM_STATUS1. See the "Functional description" section of the "Interrupt Monitor (INTM)" chapter for details. Read this field to identify the reason for a fault in case of FCCU NCF 6. 0b - No 1b - Yes
1 INTM_0_ERR	INTM_0 Error Specifies whether INTM_0 reported an error. The reported error is recorded in INTM.INTM_STATUS0. See the "Functional description" section of the "Interrupt Monitor (INTM)" chapter for details. Read this field to identify the reason for a fault in case of FCCU NCF 6. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 TEST_ACTIVATION_3_ERR	Accidental Partial Test Activation 3 Error Specifies whether partial test 3 is activated accidentally. Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No 1b - Yes

37.2.6 Read-Only GPR On Functional Reset 1 (DCMROF1)

Offset

Register	Offset
DCMROF1	300h

Function

Specifies current transfer channel ID.

This register resets after functional reset 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										Reserved	Reserved	Reserved	Reserved	EMAC_MD...	EMAC_MD...
W	W1C										W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 EMAC_MDC_CHID_1	EMAC eDMA Channel ID1 Status Specifies whether channel ID1 is the current transfer channel ID. 0b - No 1b - Yes
0 EMAC_MDC_CHID_0	EMAC eDMA Channel ID0 Status Specifies whether channel ID0 is the current transfer channel ID. 0b - No 1b - Yes

37.2.7 Read-Only GPR On Functional Reset 2 (DCMROF2)

Offset

Register	Offset
DCMROF2	304h

Function

Specifies the SDID0 contents that DCM scans from the flash memory.

This register resets after functional reset 2.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCF_SDID0															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCF_SDID0															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DCF_SDID0	DCF Client SDID 0 Configuration

37.2.8 Read-Only GPR On Functional Reset 3 (DCMROF3)

Offset

Register	Offset
DCMROF3	308h

Function

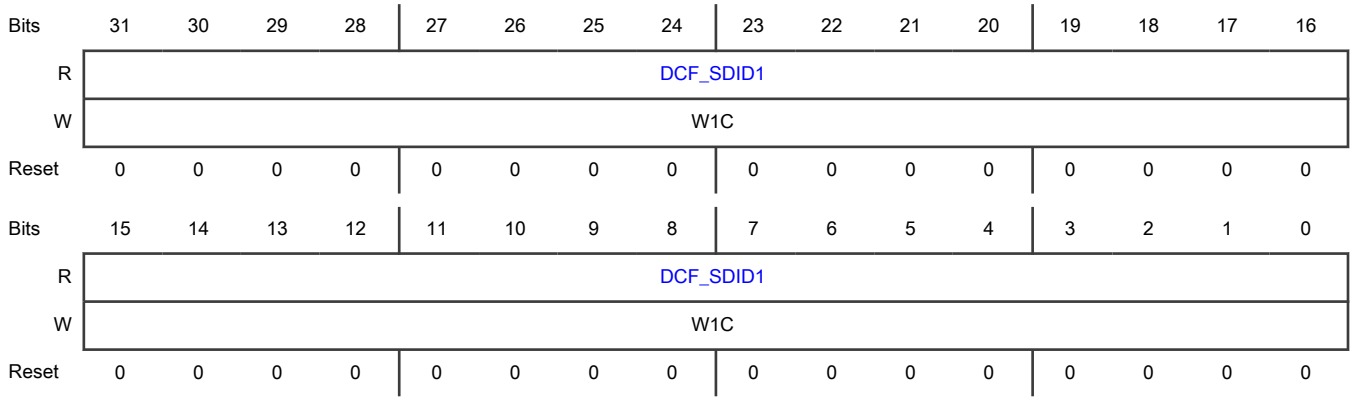
Specifies the SDID1 contents that DCM scans from the flash memory.

This register resets after functional reset 3.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID1	DCF Client SDID 1 Configuration

37.2.9 Read-Only GPR On Functional Reset 4 (DCMROF4)

Offset

Register	Offset
DCMROF4	30Ch

Function

Specifies the SDID2 contents that DCM scans from the flash memory.

This register resets after functional reset 4.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCF_SDID2															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCF_SDID2															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DCF_SDID2	DCF Client SDID 2 Configuration

37.2.10 Read-Only GPR On Functional Reset 5 (DCMROF5)

Offset

Register	Offset
DCMROF5	310h

Function

Specifies the SDID3 contents that DCM scans from the flash memory.

This register resets after functional reset 5.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCF_SDID3															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCF_SDID3															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DCF_SDID3	DCF Client SDID 3 Configuration

37.2.11 Read-Only GPR On Functional Reset 6 (DCMROF6)

Offset

Register	Offset
DCMROF6	314h

Function

Specifies the SDID4 contents that DCM scans from the flash memory.

This register resets after functional reset 6.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCF_SDID4															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCF_SDID4															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DCF_SDID4	DCF Client SDID 4 Configuration

37.2.12 Read-Only GPR On Functional Reset 7 (DCMROF7)

Offset

Register	Offset
DCMROF7	318h

Function

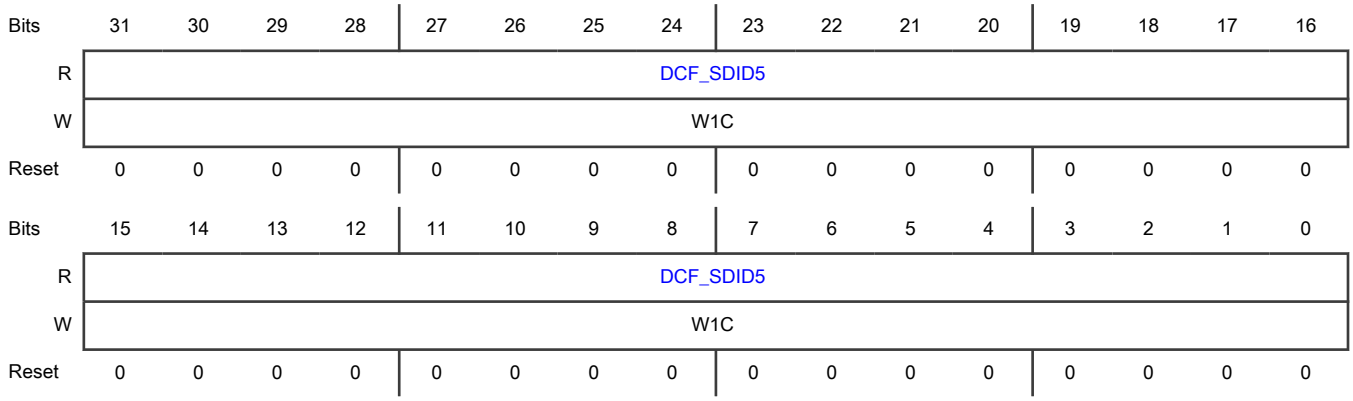
Specifies the SDID5 contents that DCM scans from the flash memory.

This register resets after functional reset 7.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID5	DCF Client SDID 5 Configuration

37.2.13 Read-Only GPR On Functional Reset 8 (DCMROF8)

Offset

Register	Offset
DCMROF8	31Ch

Function

Specifies the SDID6 contents that DCM scans from the flash memory.

This register resets after functional reset 8.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCF_SDID6															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCF_SDID6															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DCF_SDID6	DCF Client SDID 6 Configuration

37.2.14 Read-Only GPR On Functional Reset 9 (DCMROF9)

Offset

Register	Offset
DCMROF9	320h

Function

Specifies the SDID7 contents that DCM scans from the flash memory.

This register resets after functional reset 9.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCF_SDID7															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCF_SDID7															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DCF_SDID7	DCF Client SDID 7 Configuration

37.2.15 Read-Only GPR On Functional Reset 10 (DCMROF10)

Offset

Register	Offset
DCMROF10	324h

Function

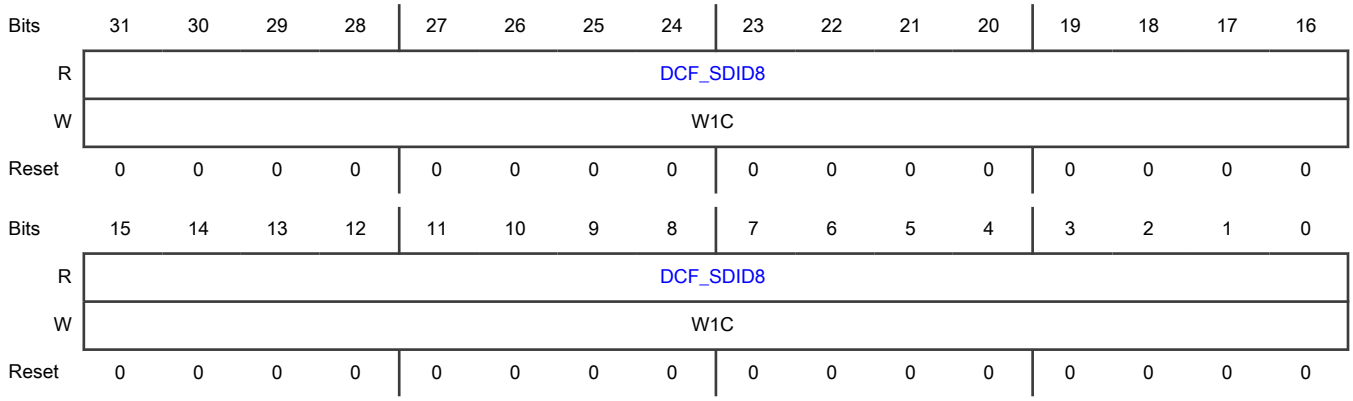
Specifies the SDID8 contents that DCM scans from the flash memory.

This register resets after functional reset 10.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID8	DCF Client SDID 8 Configuration

37.2.16 Read-Only GPR On Functional Reset 11 (DCMROF11)

Offset

Register	Offset
DCMROF11	328h

Function

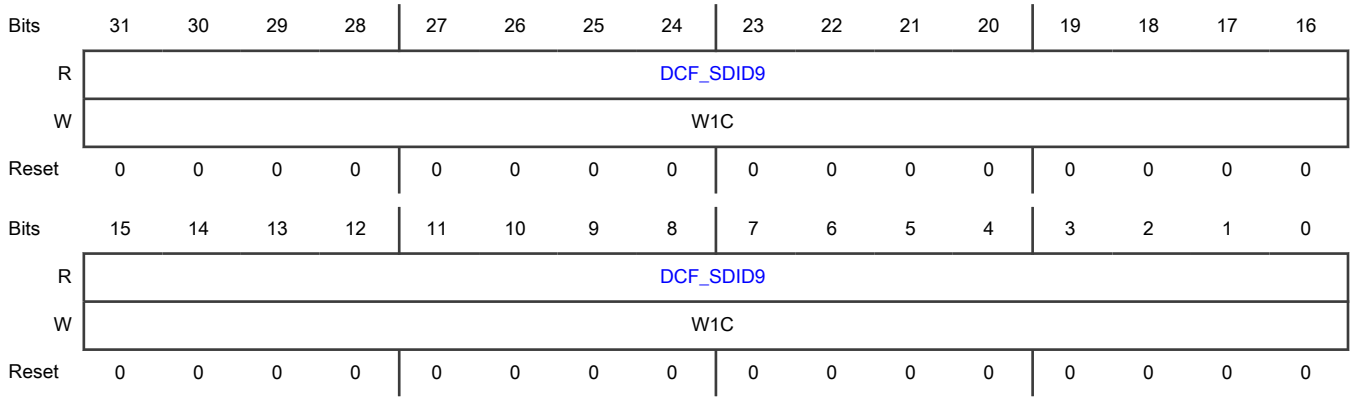
Specifies the SDID9 contents that DCM scans from the flash memory.

This register resets after functional reset 11.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID9	DCF Client SDID 9 Configuration

37.2.17 Read-Only GPR On Functional Reset 12 (DCMROF12)

Offset

Register	Offset
DCMROF12	32Ch

Function

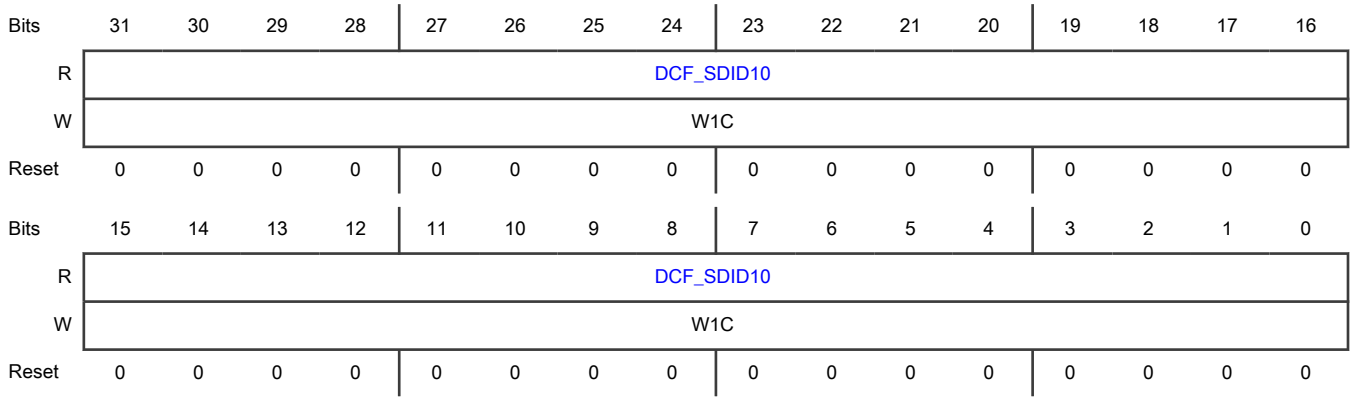
Specifies the SDID10 contents that DCM scans from the flash memory.

This register resets after functional reset 12.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID10	DCF Client SDID 10 Configuration

37.2.18 Read-Only GPR On Functional Reset 13 (DCMROF13)

Offset

Register	Offset
DCMROF13	330h

Function

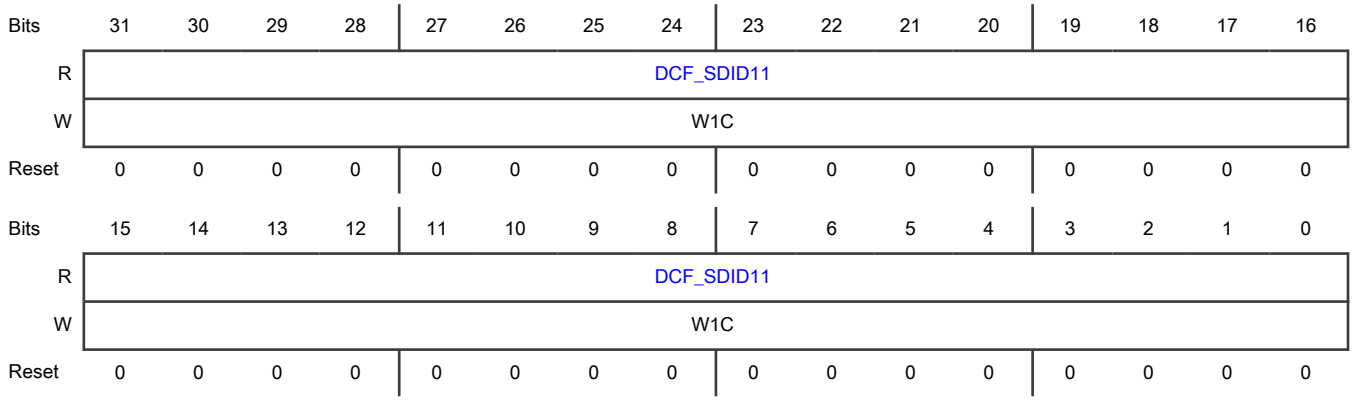
Specifies the SDID11 contents that DCM scans from the flash memory.

This register resets after functional reset 13.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID11	DCF Client SDID 11 Configuration

37.2.19 Read-Only GPR On Functional Reset 14 (DCMROF14)

Offset

Register	Offset
DCMROF14	334h

Function

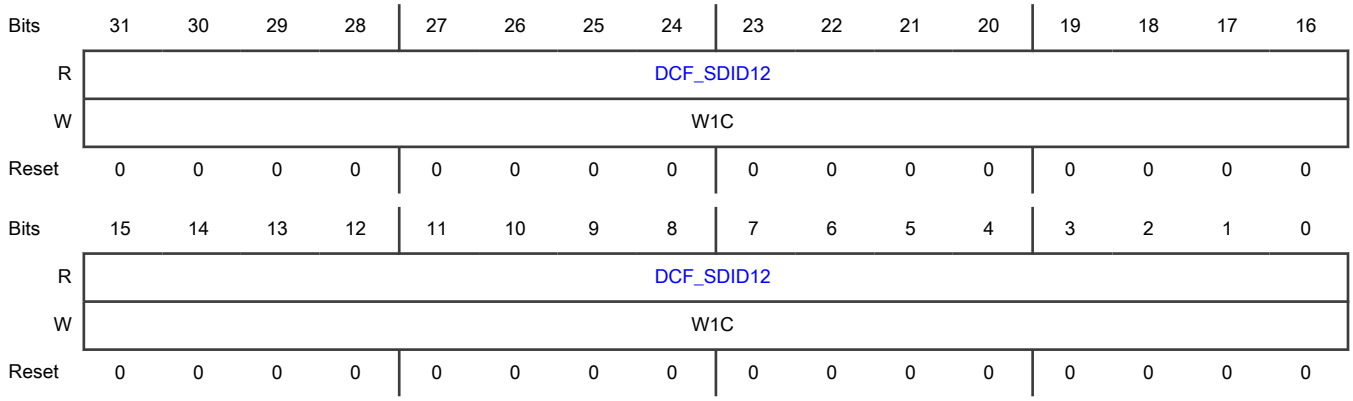
Specifies the SDID12 contents that DCM scans from flash memory.

This register resets after functional reset 14.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID12	DCF Client SDID 12 Configuration

37.2.20 Read-Only GPR On Functional Reset 15 (DCMROF15)

Offset

Register	Offset
DCMROF15	338h

Function

Specifies the SDID13 contents that DCM scans from the flash memory.

This register resets after functional reset 15.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCF_SDID13															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCF_SDID13															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DCF_SDID13	DCF Client SDID 13 Configuration

37.2.21 Read-Only GPR On Functional Reset 16 (DCMROF16)

Offset

Register	Offset
DCMROF16	33Ch

Function

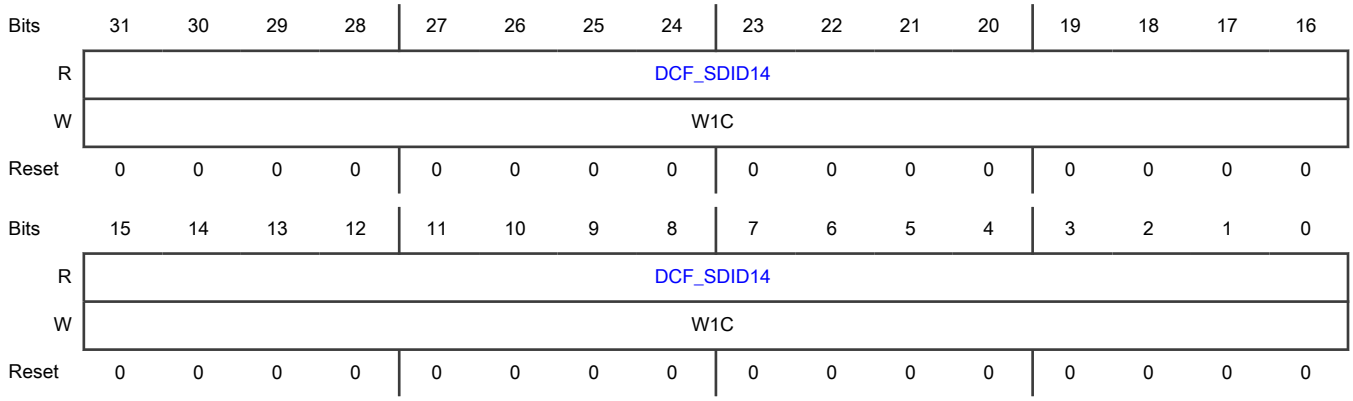
Specifies the SDID14 contents that DCM scans from the flash memory.

This register resets after functional reset 16.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID14	DCF Client SDID 14 Configuration

37.2.22 Read-Only GPR On Functional Reset 17 (DCMROF17)

Offset

Register	Offset
DCMROF17	340h

Function

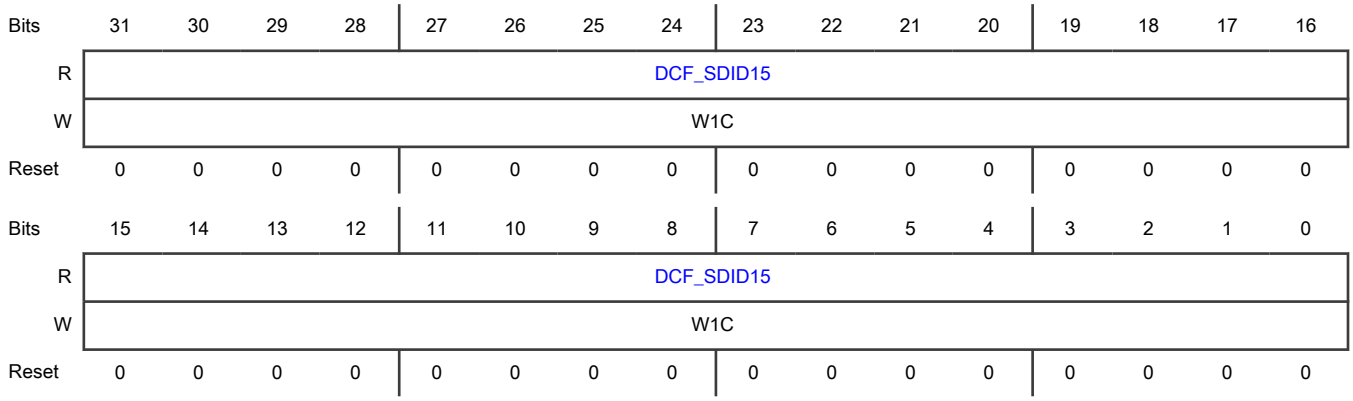
Specifies the SDID15 contents that DCM scans from the flash memory.

This register resets after functional reset 17.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID15	DCF Client SDID 15 Configuration

37.2.23 Read-Only GPR On Functional Reset 19 (DCMROF19)

Offset

Register	Offset
DCMROF19	348h

Function

Contains information related to:

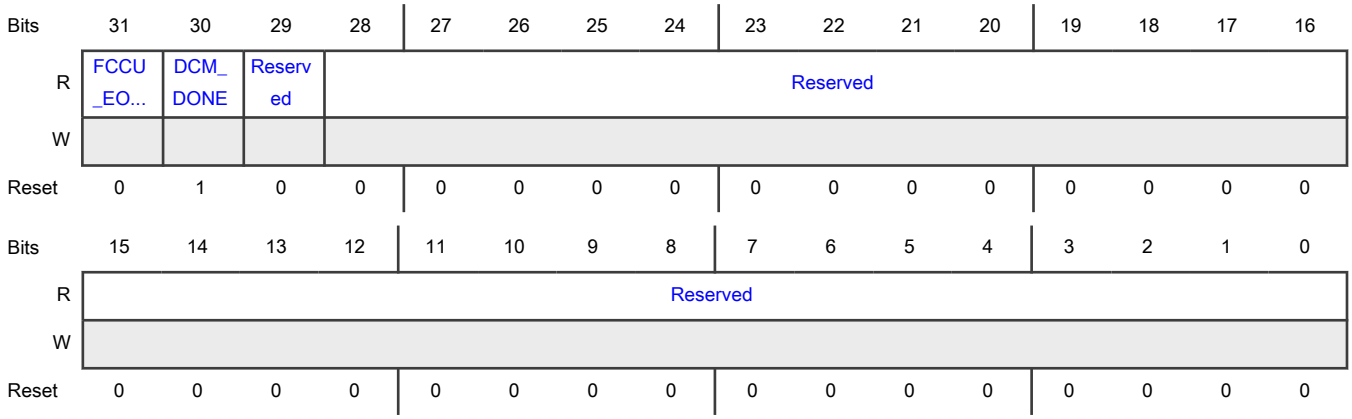
- FCCU EOUT status.
- Flash memory scanning status.
- Lockstep enable.

This register resets after functional reset 19.

NOTE

The reset value is undefined on reset and is loaded from the flash memory contents at the end of the reset sequence.

Diagram



Fields

Field	Function
31 FCCU_EOUT_DEDICATED	FCCU EOUT Status Specifies the status of FCCU_EOUT pins on GPIO_2 and GPIO_3 as configured in the DCF record, UTEST_MISC[FCCU_EOUT_DEDICATED]. See the DCF clients file attached to this document for more information. 0b - General purpose, supporting all functions 1b - Dedicated EOUT pins
30 DCM_DONE	Flash Memory Scanning Status Specifies the status of flash memory scanning by DCM. 0b - Incomplete 1b - Complete
29 —	Reserved
28-0 —	Reserved

37.2.24 Read-Only GPR On Functional Reset 20 (DCMROF20)

Offset

Register	Offset
DCMROF20	34Ch

Function

Specifies the information of chip destructive reset escalation support for destructive reset sources as configured in the DCF record, DEST_RST_ESC[13:0]. See the DCF clients file attached to this document for the mapping of corresponding destructive reset events.

This register resets after functional reset 20.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				Reserved					AIPS_IA...	QSPI_IA...	Reserved	DMA_AXB...	Reserved	LMAUTO_...	POR_WDG...
W																
Reset	0	0	0	0	0	0	0	0	0	u[2]	u[1]	0	u[2]	0	u[1]	u[1]

1. The reset value of this register is dependent on the DCF client's default value.
2. The reset value of this register is dependent on DCF client default value.

Fields

Field	Function
31-18 DCF_DEST_RST_ESC	DCF Destructive Reset Escalation Enables the destructive reset escalation feature for the corresponding destructive reset event. 00_0000_0000_0000b - Disables 00_0000_0000_0001b - Enables
17 —	Reserved
16 —	Reserved
15-12 —	Reserved
11-7 —	Reserved
6	Status of AIPS1/2 IAHB gasket as configured in DCF record, UTEST_MISC[AIPS_IAHB_BYP].

Table continues on the next page...

Table continued from the previous page...

Field	Function
AIPS_IAHB_BYP	0b - Register wall enabled. 1b - Register wall bypassed.
5 QSPI_IAHB_BYP	QuadSPI IAHB Bypass Status Specifies the status of the QuadSPI IAHB gasket as configured in the DCF record, UTEST_MISC[QSPI_IAHB_BYP]. See the DCF clients file attached to this document for more information. 0b - Register wall enabled 1b - Register wall bypassed
4 —	Reserved
3 DMA_AXBS_IAHB_BYP	Status of DMA AXBS IAHB gasket as configured in DCF record, UTEST_MISC[DMA_AXBS_IAHB_BYP]. 0b - Register wall enabled. 1b - Register wall bypassed.
2 —	Reserved
1 LMAUTO_DIS	Last Mile Auto Support Specifies whether the PMC last-mile automatic crossover from the boot regulation feature is supported for the chip. 0b - Yes 1b - No
0 POR_WDG_EN	POR Watchdog (POR_WDG) Status Specifies the status of POR_WDG as configured in the DCF record, UTEST_MISC[POR_WDG_EN]. The POR_WDG is enabled by default. See the DCF clients file attached to this document for more information. 0b - Disabled 1b - Enabled

37.2.25 Read-Only GPR On Functional Reset 21 (DCMROF21)

Offset

Register	Offset
DCMROF21	350h

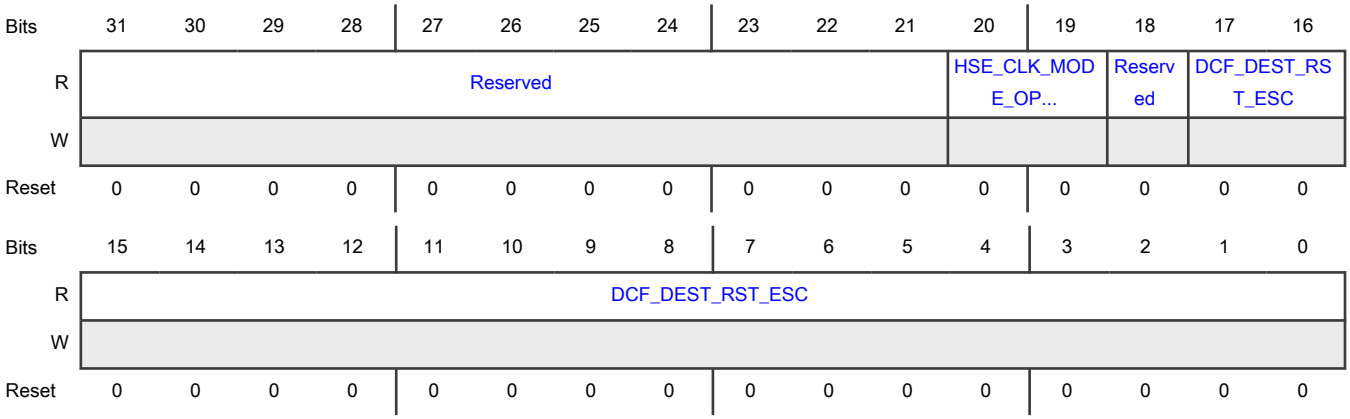
Function

Specifies the information of chip destructive reset escalation support for destructive reset sources as configured in the DCF record, DEST_RST_ESC[31:14].

This register resets after functional reset 21.

See the DCF clients file attached to this document for the mapping of corresponding destructive reset events.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-19 HSE_CLK_MODE_OPTION	HSE_B Clock Mode Option Specifies the applicable clocking options. • If the value of this field is 0b, the ratio of 1:2 between the HSE_B IPS interface clock (AIPS_SLOW_CLK), HSE_B module clock (HSE_CLK), and HSE_IAHB gasket is enabled. • If the value of this field is 1b, the ratio of 1:2 between the HSE_B IPS interface clock (AIPS_SLOW_CLK), HSE_B module clock (HSE_CLK), and HSE_IAHB gasket is bypassed. • If the value of this field is 10b or 11b, the ratio of 1:4 between the HSE_B IPS interface clock (AIPS_SLOW_CLK), HSE_B module clock (HSE_CLK), and HSE_IAHB gasket is enabled. 00b - Option A 01b - Options C, D, E, E2, and F 10b - Option B 11b - Option B
18 —	Reserved
17-0	DCF Destructive Reset Escalation Enables the destructive reset escalation feature for the corresponding destructive reset event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
DCF_DEST_RST_ESC	00_0000_0000_0000_0000b - Disables 00_0000_0000_0000_0001b - Enables

37.2.26 Read Write GPR On POR 1 (DCMRWP1)

Offset

Register	Offset
DCMRWP1	400h

Function

Contains information related to:

- Voltage dividers.
- Supply voltage monitoring.
- Ethernet modes.
- Software NCFs.

Diagram

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

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

Fields

Field	Function
31-24 —	Reserved
23-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-11 —	Reserved
10-9 POR_WDOG_TRIM	POR_WDG Trim Specifies the trims for the POR_WDG timeout value. 00b - POR_WDG timeout = 06.25 ms 01b - POR_WDG timeout = 12.50 ms 10b - POR_WDG timeout = 25.00 ms 11b - POR_WDG timeout = 50.00 ms
8 STANBDY_PWDOG_DIS	Standby POR_WDG Disable Disables the standby entry and exit monitoring window of POR_WDG. 0b - Enables 1b - Disables
7 —	Reserved
6 —	Reserved
5-4 —	Reserved
3 CLKOUT_STANDBY	Clockout Standby Expose Over Functional And Destructive Reset Specifies whether the CLKOUT_STANDBY function is available during functional or destructive reset on PTA12. 0b - No 1b - Yes
2-1 —	Reserved
0 —	Reserved

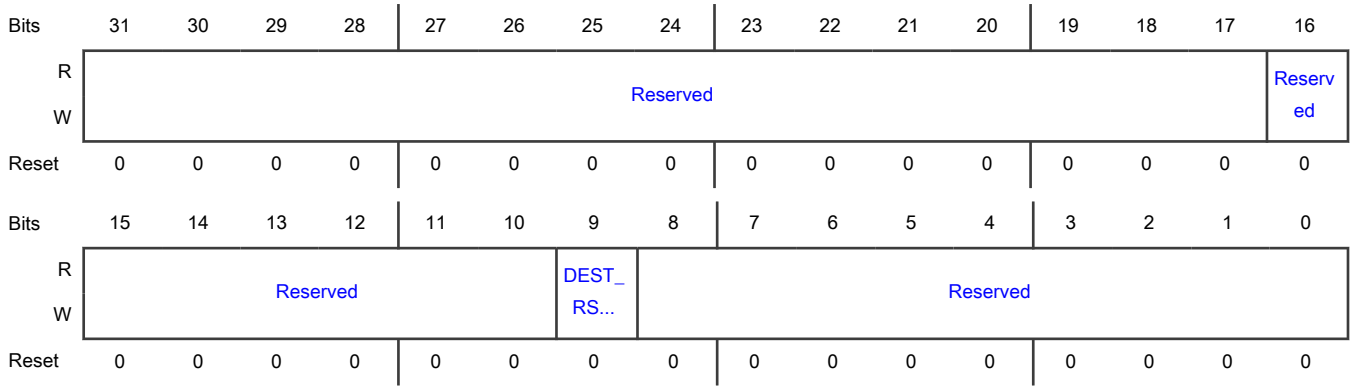
37.2.27 Read Write GPR On POR 3 (DCMRWP3)

Offset

Register	Offset
DCMRWP3	408h

Function
Resets after POR 3.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 —	Reserved
15-10 —	Reserved
9 DEST_RST9_AS_IPI	Destructive Reset 9 Configures a destructive reset to interrupt. 0b - Destructive reset 1b - PLL LOL interrupt
8-0 —	Reserved

37.2.28 Read Write GPR On Destructive Reset 2 (DCMRWD2)

Offset

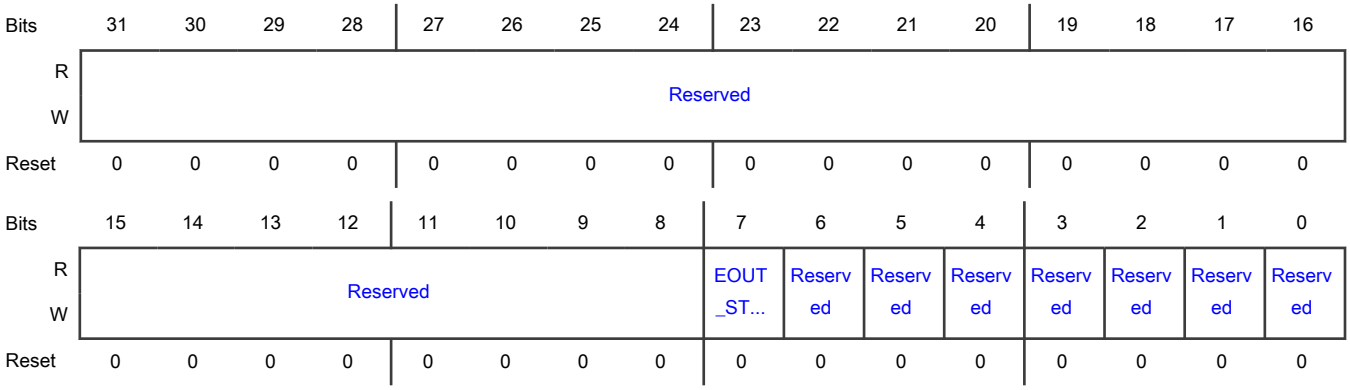
Register	Offset
DCMRWD2	504h

Function

Controls the EOUT state during self-test.

This register resets after destructive reset 2.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 EOUT_STAT_DUR_STEST	Controls the EOUT state during self-test. If this field = 0, the EOUT state changes to high impedance post self-test when the chip is under reset, and if this field = 1, the EOUT state remains in Fault state until this field becomes 0. 0b - High impedance 1b - Fault state
6 —	Reserved
5 —	Reserved
4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

37.2.29 Read Write GPR On Destructive Reset 3 (DCMRWD3)

Offset

Register	Offset
DCMRWD3	508h

Function
Includes fault disable fields and resets after destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CM7_1_I...	CM7_0_I...	CM7_1_D...	CM7_0_D...	CM7_1_D...	CM7_0_D...	PRAM0_E...	PRAM1_E...	Reserved	LC_ERR_...	Reserved		PERIPH_...	EMAC_GS...	TCM_AXB...	DATA_ED...
Reset	1	1	1	1	1	1	1	1	0	1	0	0	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR_ED...	AIPS2_G...	AIPS1_G...	QSPI_GS...	HSE_GSK...	Reserved	DMA_AXB...	SYS_AXB...	DMA_PER...	DMA_SYS...	TCM_GSK...	Reserved	Reserved	Reserved	CM7_1_L...	CM7_0_L...
Reset	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31	Cortex-M7_1 I-cache ECC Error Enable Specifies whether the Cortex-M7_1 core's I-cache data memory detected a multi-bit ECC error.

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_1_ICDATA_ECC_ERR_EN	This field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_1 core's I-cache data memory. 0b - No 1b - Yes
30 CM7_0_ICDATA_ECC_ERR_EN	Cortex-M7_0 I-cache ECC Error Enable Specifies whether the Cortex-M7_0 core's I-cache data memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_0 core's I-cache data memory. 0b - No 1b - Yes
29 CM7_1_DCTAG_ECC_ERR_EN	Cortex-M7_1 D-cache Tag ECC Error Enable Specifies whether the Cortex-M7_1 core's D-cache tag memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_1 core's D-cache tag memory. 0b - No 1b - Yes
28 CM7_0_DCTAG_ECC_ERR_EN	Cortex-M7_0 D-cache Tag ECC Error Enable Specifies whether the Cortex-M7_0 core's D-cache tag memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_0 core's D-cache tag memory. 0b - No 1b - Yes
27 CM7_1_DCDATA_ECC_ERR_EN	Cortex-M7_1 D-cache Data ECC Error Enable Specifies whether the Cortex-M7_1 core's D-cache data memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_1 core's D-cache data memory. 0b - No 1b - Yes
26 CM7_0_DCDATA_ECC_ERR_EN	Cortex-M7_0 D-cache Data ECC Error Enable Specifies whether the Cortex-M7_0 core's D-cache data memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_0 core's D-cache data memory. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
25 PRAM0_ECC_ERR_EN	PRAM0 ECC Error Enable Specifies whether a multi-bit ECC error occurred from PRAM0. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from PRAM0. 0b - No 1b - Yes
24 PRAM1_ECC_ERR_EN	PRAM1 ECC Error Enable Specifies whether a multi-bit ECC error occurred from PRAM1. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from PRAM1. 0b - No 1b - Yes
23 —	Reserved
22 LC_ERR_EN	Life Cycle Scanning Error Enable Specifies whether an error is encountered during life-cycle scanning. The field enables fault monitoring at FCCU NCF 3 if there is an error in life-cycle scanning. **NOTE** On any POR or destructive reset event, because this field becomes 0, the field has no effect. A life-cycle error (in case it is present) is not disabled. 0b - No 1b - Yes
21-20 —	Reserved
19 PERIPH_AXBS_ALARM_EN	Peripheral AXBS Alarm Enable Specifies whether peripheral AXBS_Lite reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 in case of a peripheral AXBS_Lite safety alarm. 0b - No 1b - Yes
18 EMAC_GSKT_ALARM_EN	EMAC Gasket Alarm Enable Specifies whether the EMAC IAHB gasket reported an alarm. The field enables fault monitoring at FCCU NCF 1 in case of an EMAC IAHB gasket alarm. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 TCM_AXBS_ALARM_EN	TCM AXBS Alarm Enable Specifies whether the TCM AHB splitter reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 in case of a TCM AHB splitter safety alarm. 0b - No 1b - Yes
16 DATA_EDC_ERR_EN	Data EDC Error Enable Specifies whether a data EDC error occurred. The field enables fault monitoring at FCCU NCF 1, in case of an integrity error on data, for safety. 0b - No 1b - Yes
15 ADDR_EDC_ERR_EN	Address EDC Error Enable Specifies whether an address integrity (EDC) error occurred. The field enables fault monitoring at FCCU NCF 1, in case of an integrity error on address, for safety. 0b - No 1b - Yes
14 AIPS2_GSKT_ALARM_EN	Enable bit for enabling the fault monitoring at FCCU NCF 1 for the fault: AIPS2 IAHB gasket alarm. 0b - No alarm indicated by AIPS2 IAHB gasket. 1b - Alarm indicated by AIPS2 IAHB gasket.
13 AIPS1_GSKT_ALARM_EN	Enable bit for enabling the fault monitoring at FCCU NCF 1 for the fault: AIPS1 IAHB gasket alarm. 0b - No alarm indicated by AIPS1 IAHB gasket. 1b - Alarm indicated by AIPS1 IAHB gasket.
12 QSPI_GSKT_ALARM_EN	QuadSPI Gasket Alarm Enable Specifies whether the QuadSPI IAHB gasket reported an alarm. The field enables fault monitoring at FCCU NCF 1 in case of QuadSPI IAHB gasket alarm. 0b - No 1b - Yes
11 HSE_GSKT_ALARM_EN	HSE_B Gasket Alarm Enable Specifies whether the HSE_B IAHB gasket reported an alarm. The field enables fault monitoring at FCCU NCF 1 in case of HSE_B IAHB gasket alarm. 0b - No 1b - Yes
10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 DMA_AXBS_ALARM_EN	DMA AXBS Alarm Enable Specifies whether eDMA AXBS_Lite reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 in case of an eDMA AXBS_Lite safety alarm. 0b - No 1b - Yes
8 SYS_AXBS_ALARM_EN	System AXBS Alarm Enable Specifies whether the system AXBS reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 in case of a system AXBS safety alarm. 0b - No 1b - Yes
7 DMA_PERIPH_GSKT_ALARM_EN	TCM Gasket Alarm Enable Specifies whether the eDMA-peripheral AXBS IAHB gasket reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 in case of IAHB gasket safety alarm from the eDMA peripheral AXBS IAHB gasket. 0b - No 1b - Yes
6 DMA_SYS_GSKT_ALARM_EN	DMA System Gasket Alarm Enable Specifies whether the eDMA-system AXBS IAHB gasket reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 in case of IAHB gasket safety alarm from the eDMA-system AXBS IAHB gasket. 0b - No 1b - Yes
5 TCM_GSKT_ALARM_EN	TCM Gasket Alarm Enable Specifies whether the TCM IAHB gasket reported an alarm. If this field = 1, the gasket reports a monitor alarm. The field enables fault monitoring at FCCU NCF 1 in case of TCM IAHB gasket monitor alarm. 0b - No 1b - Yes
4 —	Reserved
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 —	Reserved
1 CM7_1_LOCKUP_EN	Cortex-M7_1 Lockup Enable Specifies whether the Cortex-M7_1 core is in the Lockup state. The field enables fault monitoring at FCCU NCF 0 in case of the Cortex-M7_1 core lockup. 0b - No 1b - Yes
0 CM7_0_LOCKUP_EN	Cortex-M7 Lockup Enable Specifies whether the Cortex-M7_0 core is in the Lockup state. The field enables fault monitoring at FCCU NCF 0 in case of the Cortex-M7_0 core lockup. 0b - No 1b - Yes

37.2.30 Read Write GPR On Destructive Reset 4 (DCMRWD4)

Offset

Register	Offset
DCMRWD4	50Ch

Function

Contains information related to:

- Test activation errors.
- Fault monitoring.
- DCM flash memory scanning.
- ECC errors.

This register resets after destructive reset 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	TEST_AC...	TEST_AC...	Reserved	VDD2P5_...	VDD1P1_...	Reserved	FLASH_A...	Reserved	FLASH_S...	FLASH_R...	FLASH_R...	FLASH_A...	FLASH_E...	PF2_DAT...	PF2_COD...
Reset	1	1	1	0	1	1	1	1	0	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PF1_DAT...	PF1_COD...	PF0_DAT...	PF0_COD...	Reserved	PRAM1_F...	PRAM0_F...	DMA_TCD...	CM7_1_D...	CM7_1_D...	CM7_1_I...	CM7_0_D...	CM7_0_D...	CM7_0_I...	CM7_1_I...	CM7_0_I...
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 —	Reserved
30 TEST_ACTIVATION_1_ERR_EN	Test Activation 1 Error Enable Specifies whether a partial test is activated accidentally. The field enables fault monitoring at FCCU NCF 5 in case of an accidental partial test activation 1. 0b - No 1b - Yes
29 TEST_ACTIVATION_0_ERR_EN	Test Activation 0 Error Enable Specifies whether a partial test is activated accidentally. The field enables fault monitoring at FCCU NCF 5 in case of an accidental partial test activation 0. 0b - No 1b - Yes
28 —	Reserved
27 VDD2P5_GNG_ERR_EN	VDD2P5 Go/No-go Error Enable Specifies whether the VDD2P5 (double bond) supply is clean. The field enables fault monitoring at FCCU NCF 4 if there is a "go/no-go" indicator for VDD_HV_FLA (double bond) going to FXOSC and PLL. If the value of this field = 0, there is a "go" indication that the supply if clean; if it = 1, there is a "no-go" indication that the supply is unclean and there is a fault in double-bond connection or its routing within the chip. 0b - Clean 1b - Unclean
26	VDD1PD1 Go/No-go Error Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
VDD1P1_GNG_ERR_EN	Specifies whether the VDD1PD1 (double bond) supply is clean. The field enables fault monitoring at FCCU NCF 4 if there is a "go/no-go" indicator for VDD1PD1 (double bond) supply going to PLL. If the value of this field = 0, there is a "go" indication that the supply if clean; if it = 1, there is a "no-go" indication that the supply is unclean and there is a fault in double-bond connection or its routing within the chip. 0b - Clean 1b - Unclean
25 —	Reserved
24 FLASH_ACCESS_ERR_EN	Flash Memory Access Error Enable Specifies whether a transaction monitor mismatch error occurred from the flash memory controller. The field enables fault monitoring at FCCU NCF 3 in case of a transaction monitor mismatch error from the flash memory controller. The alarm indicates that the flash memory controller detected a transaction monitor mismatch when compared to the flash memory safety feedback output. The flash memory specifies where the reconstructed address is compared with the address that invoked the flash memory access. 0b - No 1b - Yes
23 —	Reserved
22 FLASH_SCAN_ERR_EN	Flash Memory Scanning Error Enable Specifies whether an error occurred during DCM flash memory scanning. The field enables fault monitoring at FCCU NCF 3 in case of an error during the DCM flash memory scanning process because of invalid data. **NOTE** On a POR or destructive reset event, because this field becomes 0, the field has no effect. A life-cycle error (in case it is present) is not disabled. 0b - No 1b - Yes
21 FLASH_RST_ERR_EN	Flash Memory Reset Error Enable Specifies whether a flash memory reset error occurred. The field enables fault monitoring at FCCU NCF 3 in case of a flash memory reset error. This error indication is set when the flash memory encounters errors during its reset reads. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 FLASH_REF_ERR_EN	Flash Memory Reference Error Encode Specifies whether a reference current loss or read voltage error occurred during previous read(s). The field enables fault monitoring at FCCU NCF 3 if there is a flash memory reference current loss or read voltage error during previous read(s). 0b - No 1b - Yes
19 FLASH_ADDR_ENC_ERR_EN	Flash Memory Address Encode Error Enable Specifies whether an address encode error occurred in the flash memory. The field enables fault monitoring at FCCU NCF 3 if there is a flash memory address encode error. During address decoding, if multiple or no address line is selected, FMU reports an address encode error. 0b - No 1b - Yes
18 FLASH_EDC_ERR_EN	Flash Memory EDC Error Enable Specifies whether EDC after ECC error is reported in the flash memory. The field enables fault monitoring at FCCU NCF 3 in case of a flash memory ECC correction error via EDC, reported by FMU. 0b - No 1b - Yes
17 PF2_DATA_ECC_ERR_EN	PF2 Data ECC Error Enable Specifies whether FMU reported an uncorrectable error in the flash memory controller port 2 data memory. The field enables fault monitoring at FCCU NCF 3 in case of Flash2 data ECC uncorrectable error. 0b - No 1b - Yes
16 PF2_CODE_ECC_ERR_EN	PF2 Code ECC Error Enable Specifies whether FMU reported an uncorrectable error in the flash memory controller port 2 code memory. The field enables fault monitoring at FCCU NCF 3 in case of Flash2 code ECC uncorrectable error. 0b - No 1b - Yes
15 PF1_DATA_ECC_ERR_EN	PF1 Data ECC Error Enable Specifies whether FMU reported an uncorrectable error in the flash memory controller port 1 data memory. The field enables fault monitoring at FCCU NCF 3 in case of Flash1 data ECC uncorrectable error. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 PF1_CODE_ECC_ERR_EN	PF1 Code ECC Error Enable Specifies whether FMU reported an uncorrectable error in the flash memory controller port 1 code memory. The field enables fault monitoring at FCCU NCF 3 in case of Flash1 code ECC uncorrectable error. 0b - No 1b - Yes
13 PF0_DATA_ECC_ERR_EN	PF0 Data ECC Error Enable Specifies whether FMU reported an uncorrectable error in the flash memory controller port 0 data memory. The field enables fault monitoring at FCCU NCF 3 in case of Flash0 data ECC uncorrectable error. 0b - No 1b - Yes
12 PF0_CODE_ECC_ERR_EN	PF0 Code ECC Error Enable Specifies whether FMU reported an uncorrectable error in the flash memory controller port 0 code memory. The field enables fault monitoring at FCCU NCF 3 in case of Flash0 code ECC uncorrectable error. 0b - No 1b - Yes
11 —	Reserved
10 PRAM1_FCCU_ALARM_EN	PRAM1 FCCU Alarm Enable Specifies whether PRAM1 reported a safety alarm. The field enables fault monitoring at FCCU NCF 2 in case of PRAM1 safety alarm. This alarm is set on faulty SRAM1 read or read-modify error. 0b - No 1b - Yes
9 PRAM0_FCCU_ALARM_EN	PRAM0 FCCU Alarm Enable Specifies whether PRAM0 reported a safety alarm. The field enables fault monitoring at FCCU NCF 2 in case of PRAM0 safety alarm. This alarm is set to faulty SRAM0 read or read-modify error. 0b - No 1b - Yes
8 DMA_TCD_RAM_ECC_ERR_EN	eDMA TCD RAM ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error reported from the eDMA_TCD memory. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. 0b - No 1b - Yes
7 CM7_1_DTCM1_ECC_ERR_EN	Cortex-M7_1 DTCM 1 ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_1 core's DTCM block 1. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_1 core's DTCM 1 consists of two physical blocks. 0b - No 1b - Yes
6 CM7_1_DTCM0_ECC_ERR_EN	Cortex-M7_1 DTCM 0 ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_1 core's DTCM consists of two physical blocks. This field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_1 core's DTCM block 0. 0b - No 1b - Yes
5 CM7_1_ITCM_ECC_ERR_EN	Cortex-M7_1 ITCM ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_1 core's ITCM. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. 0b - No 1b - Yes
4 CM7_0_DTCM1_ECC_ERR_EN	Cortex-M7_0 DTCM 1 ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_0 core's DTCM block 1. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_0 core's DTCM consists of two physical blocks. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 CM7_0_DTCM0_ECC_ERR_EN	Cortex-M7_0 DTCM 0 ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_0 core's DTCM block 0. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_0 core's DTCM consists of two physical blocks. 0b - No 1b - Yes
2 CM7_0_ITCM_ECC_ERR_EN	Cortex-M7 ITCM ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_0 core's ITCM. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. 0b - No 1b - Yes
1 CM7_1_ICTAG_ECC_ERR_EN	Cortex-M7_1 I-cache Tag ECC Error Enable Specifies whether the Cortex-M7_1 core's I-cache tag memory detected a multi-bit ECC error. This field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_1 core's I-cache tag memory. 0b - No 1b - Yes
0 CM7_0_ICTAG_ECC_ERR_EN	Cortex-M7_0 I-cache Tag ECC Error Enable Specifies whether the Cortex-M7_0 core's I-cache tag memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_0 core's I-cache tag memory. 0b - No 1b - Yes

37.2.31 Read Write GPR On Destructive Reset 5 (DCMRWD5)

Offset

Register	Offset
DCMRWD5	510h

Function

Contains information related to:

- Uncorrectable ECC error detection.
- I-cache and D-cache ECC errors.
- Fault monitoring.

This register resets after destructive reset 5.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved									Reserved	CM7_0_A...	CM7_0_A...	CM7_1_A...	CM7_1_A...	DMA_RDA...	Reserved
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EMAC_RD...	TCM_RDA...	DEBUG_A...	MCT_BUS...	STCU_BI...	MBIST_A...	STCU_NC...	SW_NCF_...	SW_NCF_...	SW_NCF_...	SW_NCF_...	INTM_3_...	INTM_2_...	INTM_1_...	INTM_0_...	Reserved
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-23 —	Reserved
22 —	Reserved
21 CM7_0_AHBM_RDATA_EDC_ERR_EN	Cortex-M7_0 AHBM Read Data EDC Error Enable Specifies whether an integrity error is reported on the Cortex-M7_0 core's main read data. The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the Cortex-M7_0 core's main read data, for safety. 0b - No 1b - Yes
20 CM7_0_AHBP_RDATA_EDC_ERR_EN	Cortex-M7_0 AHBP Read Data EDC Error Enable Specifies whether an integrity error is reported on the Cortex-M7_0 core's peripheral read data. The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the Cortex-M7_0 core's peripheral read data, for safety. 0b - No 1b - Yes
19	Cortex-M7_1 AHBM Read Data EDC Error Enable Specifies whether an integrity error is reported on the Cortex-M7_1 core's main read data.

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_1_AHBM_RDATA_EDC_ERR_EN	The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the Cortex-M7_1 core's main read data, for safety. 0b - No 1b - Yes
18 CM7_1_AHBP_RDATA_EDC_ERR_EN	Cortex-M7_1 AHBP Read Data EDC Error Enable Specifies whether an integrity error is reported on the Cortex-M7_1 core's peripheral read data. The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the Cortex-M7_1 core's peripheral read data, for safety. 0b - No 1b - Yes
17 DMA_RDATA_EDC_ERR_EN	eDMA Read Data EDC Error Enable Specifies whether an integrity error is reported on the eDMA read data. The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the eDMA read data, for safety. 0b - No 1b - Yes
16 —	Reserved
15 EMAC_RDATA_EDC_ERR_EN	EMAC Read Data EDC Error Enable Specifies whether an integrity error is reported on the EMAC read data. The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the EMAC read data, for safety. 0b - No 1b - Yes
14 TCM_RDATA_EDC_ERR_EN	TCM Read Data EDC Error Enable Specifies whether an integrity error is reported on the TCM read data. The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the TCM read data, for safety. 0b - No 1b - Yes
13 DEBUG_ACTIVATION_ERR_EN	Debug Activation Error Enable Specifies whether unintended debug is activated.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The field enables fault monitoring at FCCU NCF 5 for monitoring of unintended debug activation. The value of this field is 1 when the core is in the Halted state with application debug not enabled or debugger request not enabled. **NOTE** While the debugger is connected, DEBUG_ACTIVATION_ERR_EN must be 0 to disable debug activation error monitoring because the debugger is intentionally connected to the chip. 0b - No 1b - Yes
12 MCT_BUS_ERR_EN	MCT Bus Error Enable Enable bit for enabling the fault monitoring at FCCU NCF 5 for the fault: Fault reported due to illegal access on MBIST controller (MCT). This fault is reported via a transfer error indication to the system. 0b - No transfer error indicated from MCT. 1b - Transfer error indicated from MCT.
11 STCU_BIST_USER_CF_EN	STCU2 BIST User CF Enable Specifies whether MBIST is enabled accidentally (a fault condition is detected in Run mode). The field enables fault monitoring at FCCU NCF 5 if MBIST is enabled accidentally. 0b - No 1b - Yes
10 MBIST_ACTIVATION_ERR_EN	MBIST Activation Error Enable Specifies whether accidental backdoor is enabled on memories. The field enables fault monitoring at FCCU NCF 5 in case of an accidental backdoor access on memories. This monitor needs to be disabled on FCCU when performing a fault injection. 0b - No 1b - Yes
9 STCU_NCF_EN	STCU2 NCF Enable Enables fault monitoring at FCCU NCF 5 for STCU2 NCF, that is, BIST result error. 0b - Disables 1b - Enables
8 SW_NCF_3_EN	Software NCF 3 Enable Enables fault monitoring at FCCU NCF 7 for software NCF3 (DCMRWF1[FCCU_SW_NCF3]). 0b - Disables 1b - Enables

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 SW_NCF_2_EN	Software NCF 2 Enable Enables fault monitoring at FCCU NCF 7 for software NCF2 (DCMRWF1[FCCU_SW_NCF2]). 0b - Disables 1b - Enables
6 SW_NCF_1_EN	Software NCF 1 Enable Enables fault monitoring at FCCU NCF 7 for software NCF1 (DCMRWF1[FCCU_SW_NCF1]). 0b - Disables 1b - Enables
5 SW_NCF_0_EN	Software NCF 0 Enable Enables fault monitoring at FCCU NCF 7 for software NCF0 (DCMRWF1[FCCU_SW_NCF0]). 0b - Disables 1b - Enables
4 INTM_3_ERR_EN	INTM 3 Error Enable Specifies whether interrupt monitor 3 reported an error. The field enables fault monitoring at FCCU NCF 6 if INTM reports an interrupt monitor 3 error. This error is also reported in INTM.INTM_STATUS3. See the "Functional description" section in the INTM chapter for details. 0b - No 1b - Yes
3 INTM_2_ERR_EN	INTM 2 Error Enable Specifies whether interrupt monitor 2 reported an error. The field enables fault monitoring at FCCU NCF 6 if INTM reports an interrupt monitor 2 error. This error is also reported in INTM.INTM_STATUS2. See the "Functional description" section in the INTM chapter for details. 0b - No 1b - Yes
2 INTM_1_ERR_EN	INTM 1 Error Enable Specifies whether interrupt monitor 1 reported an error. The field enables fault monitoring at FCCU NCF 6 if INTM reports an interrupt monitor 1 error. This error is also reported in INTM.INTM_STATUS1. See the "Functional description" section in the INTM chapter for details. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 INTM_0_ERR_EN	INTM 0 Error Enable Specifies whether interrupt monitor 0 reported an error. The field enables fault monitoring at FCCU NCF 6 if INTM reports an interrupt monitor 0 error. This error is also reported in INTM.INTM_STATUS0. See the "Functional description" section in the INTM chapter for details. 0b - No 1b - Yes
0 —	Reserved

37.2.32 Read Write GPR On Destructive Reset 6 (DCMRWD6)

Offset

Register	Offset
DCMRWD6	514h

Function

Contains information related to module debug disable.

This register resets after destructive reset 6.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SAI1_DB...	SAI0_DB...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXIO_...	LPI2C1_...	LPI2C0_...	LPSPI5_...	LPSPI4_...	LPSPI3_...	LPSPI2_...	LPSPI1_...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LPSPI0_...	PIT2_DB...	PIT1_DB...	PIT0_DB...	STM1_DB...	STM0_DB...	SWT1_DB...	SWT0_DB...	RTC_DBG...	EMIOS2_...	EMIOS1_...	EMIOS0_...	LCU1_DB...	LCU0_DB...	FCCU_DB...	EDMA_DB...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	SAI1 debug disable bit for CM7_0. Set this bit 1 to disable the debug of IP.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SAI1_DBG_DIS_CM7_0	0b - SAI1 enters debug mode when CM7_0 enters debug mode. 1b - SAI1 remains functional and is not impacted when CM7_0 enters debug mode.
30 SAI0_DBG_DIS_CM7_0	SAI0 debug disable bit for CM7_0. Set this bit 1 to disable the debug of IP. 0b - SAI0 enters debug mode when CM7_0 enters debug mode. 1b - SAI0 remains functional and is not impacted when CM7_0 enters debug mode.
29 FLEXCAN5_DBG_DIS_CM7_0	FlexCAN_5 Debug Disable Cortex-M7_0 Specifies whether FlexCAN_5 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
28 FLEXCAN4_DBG_DIS_CM7_0	FlexCAN_4 Debug Disable Cortex-M7_0 Specifies whether FlexCAN_4 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
27 FLEXCAN3_DBG_DIS_CM7_0	FlexCAN_3 Debug Disable Cortex-M7_0 Specifies whether FlexCAN_3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
26 FLEXCAN2_DBG_DIS_CM7_0	FlexCAN_2 Debug Disable Cortex-M7_0 Specifies whether FlexCAN_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
25 FLEXCAN1_DBG_DIS_CM7_0	FlexCAN_1 Debug Disable Cortex-M7_0 Specifies whether FlexCAN_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enters Debug mode 1b - Remains functional and unimpacted
24 FLEXCAN0_DBG_DIS_CM7_0	FlexCAN_0 Debug Disable Cortex-M7_0 Specifies whether FlexCAN_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
23 FLEXIO_DBG_DIS_CM7_0	FlexIO Debug Disable Cortex-M7_0 Specifies whether FlexIO enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
22 LPI2C1_DBG_DIS_CM7_0	LPI2C_1 Debug Disable Cortex-M7_0 Specifies whether LPI2C_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
21 LPI2C0_DBG_DIS_CM7_0	LPI2C_0 Debug Disable Cortex-M7_0 Specifies whether LPI2C_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
20 LPSPI5_DBG_DIS_CM7_0	LPSPI_5 Debug Disable Cortex-M7_0 Specifies whether LPSPI_5 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
19	LPSPI_4 Debug Disable Cortex-M7_0

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPSPI4_DBG_DIS_CM7_0	Specifies whether LPSPI_4 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
18 LPSPI3_DBG_DIS_CM7_0	LPSPI_3 Debug Disable Cortex-M7_0 Specifies whether LPSPI_3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
17 LPSPI2_DBG_DIS_CM7_0	LPSPI_2 Debug Disable Cortex-M7_0 Specifies whether LPSPI_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
16 LPSPI1_DBG_DIS_CM7_0	LPSPI_1 Debug Disable Cortex-M7_0 Specifies whether LPSPI_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
15 LPSPI0_DBG_DIS_CM7_0	LPSPI_0 Debug Disable Cortex-M7_0 Specifies whether LPSPI_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
14 PIT2_DBG_DIS_CM7_0	PIT_2 Debug Disable Cortex-M7_0 Specifies whether PIT_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enters Debug mode 1b - Remains functional and unimpacted
13 PIT1_DBG_DIS_CM7_0	PIT_1 Debug Disable Cortex-M7_0 Specifies whether PIT_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
12 PIT0_DBG_DIS_CM7_0	PIT_0 Debug Disable Cortex-M7_0 Specifies whether PIT_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
11 STM1_DBG_DIS_CM7_0	STM_1 Debug Disable Cortex-M7_0 Specifies whether STM_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
10 STM0_DBG_DIS_CM7_0	STM_0 Debug Disable Cortex-M7_0 Specifies whether STM_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
9 SWT1_DBG_DIS_CM7_0	SWT_1 Debug Disable Cortex-M7_0 Specifies whether SWT_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
8	SWT_0 Debug Disable Cortex-M7_0

Table continues on the next page...

Table continued from the previous page...

Field	Function
SWT0_DBG_DIS_CM7_0	Specifies whether SWT_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
7 RTC_DBG_DIS_CM7_0	RTC Debug Disable Cortex-M7_0 Specifies whether RTC enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
6 EMIOS2_DBG_DIS_CM7_0	EMIOS2 debug disable bit for CM7_0. Set this bit 1 to disable the debug of IP. 0b - eMIOS2 enters debug mode when CM7_0 enters debug mode. 1b - eMIOS2 remains functional and is not impacted when CM7_0 enters debug mode.
5 EMIOS1_DBG_DIS_CM7_0	EMIOS1 debug disable bit for CM7_0. Set this bit 1 to disable the debug of IP. 0b - eMIOS1 enters debug mode when CM7_0 enters debug mode. 1b - eMIOS1 remains functional and is not impacted when CM7_0 enters debug mode.
4 EMIOS0_DBG_DIS_CM7_0	eMIOS_0 Debug Disable Cortex-M7_0 Specifies whether eMIOS_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
3 LCU1_DBG_DIS_CM7_0	LCU_1 Debug Disable Cortex-M7_0 Specifies whether LCU_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
2 LCU0_DBG_DIS_CM7_0	LCU_0 Debug Disable Cortex-M7_0 Specifies whether LCU_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enters Debug mode 1b - Remains functional and unimpacted
1 FCCU_DBG_DIS_CM7_0	FCCU Debug Disable Cortex-M7_0 Specifies whether FCCU enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
0 EDMA_DBG_DIS_CM7_0	eDMA Debug Disable Cortex-M7_0 Specifies whether eDMA enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted

37.2.33 Read Write GPR On Destructive Reset 7 (DCMRWD7)

Offset

Register	Offset
DCMRWD7	518h

Function

Contains information related to module debug disable.

This register resets after destructive reset 7.

Diagram

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

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

Fields

Field	Function
31-26 —	Reserved
25 —	Reserved
24 —	Reserved
23-21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

37.2.34 Read Write GPR On Destructive Reset 8 (DCMRWD8)

Offset

Register	Offset
DCMRWD8	51Ch

Function

Provides module debug disable information.

This register resets after destructive reset 8.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SAI1_DB...	SAI0_DB...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXIO_...	LPI2C1_...	LPI2C0_...	LPSPI5_...	LPSPI4_...	LPSPI3_...	LPSPI2_...	LPSPI1_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LPSPI0_...	PIT2_DB...	PIT1_DB...	PIT0_DB...	STM1_DB...	STM0_DB...	SWT1_DB...	SWT0_DB...	RTC_DBG...	EMIOS2_...	EMIOS1_...	EMIOS0_...	LCU1_DB...	LCU0_DB...	FCCU_DB...	EDMA_DB...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SAI1_DBG_DIS_CM7_1	SAI1 debug disable bit for CM7_1. Set this bit 1 to disable the debug of IP. 0b - SAI1 enters debug mode when CM7_1 enters debug mode. 1b - SAI1 remains functional and is not impacted when CM7_1 enters debug mode.
30 SAI0_DBG_DIS_CM7_1	SAI0 debug disable bit for CM7_1. Set this bit 1 to disable the debug of IP. 0b - SAI0 enters debug mode when CM7_1 enters debug mode. 1b - SAI0 remains functional and is not impacted when CM7_1 enters debug mode.
29 FLEXCAN5_DBG_DIS_CM7_1	FlexCAN_5 Debug Disable Cortex-M7_1 Specifies whether FlexCAN_5 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
28 FLEXCAN4_DBG_DIS_CM7_1	FlexCAN_4 Debug Disable Cortex-M7_1 Specifies whether FlexCAN_4 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
27 FLEXCAN3_DBG_DIS_CM7_1	FlexCAN_3 Debug Disable Cortex-M7_1 Specifies whether FlexCAN_3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Remains functional and unimpacted
26 FLEXCAN2_DBG_DIS_CM7_1	FlexCAN_2 Debug Disable Cortex-M7_1 Specifies whether FlexCAN_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
25 FLEXCAN1_DBG_DIS_CM7_1	FlexCAN_1 Debug Disable Cortex-M7_1 Specifies whether FlexCAN_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
24 FLEXCAN0_DBG_DIS_CM7_1	FlexCAN_0 Debug Disable Cortex-M7_1 Specifies whether FlexCAN_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
23 FLEXIO_DBG_DIS_CM7_1	FlexIO Debug Disable Cortex-M7_1 Specifies whether FlexIO enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
22 LPI2C1_DBG_DIS_CM7_1	LPI2C_1 Debug Disable Cortex-M7_1 Specifies whether LPI2C_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
21	LPI2C_0 Debug Disable Cortex-M7_1 Specifies whether LPI2C_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPI2C0_DBG_DIS_CM7_1	Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
20 LPSPI5_DBG_DIS_CM7_1	LPSPI_5 Debug Disable Cortex-M7_1 Specifies whether LPSPI_5 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
19 LPSPI4_DBG_DIS_CM7_1	LPSPI_4 Debug Disable Cortex-M7_1 Specifies whether LPSPI_4 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
18 LPSPI3_DBG_DIS_CM7_1	LPSPI_3 Debug Disable Cortex-M7_1 Specifies whether LPSPI_3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
17 LPSPI2_DBG_DIS_CM7_1	LPSPI_2 Debug Disable Cortex-M7_1 Specifies whether LPSPI_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
16 LPSPI1_DBG_DIS_CM7_1	LPSPI_1 Debug Disable Cortex-M7_1 Specifies whether LPSPI_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 LPSPI0_DBG_DIS_CM7_1	LPSPI_0 Debug Disable Cortex-M7_1 Specifies whether LPSPI_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
14 PIT2_DBG_DIS_CM7_1	PIT_2 Debug Disable Cortex-M7_1 Specifies whether PIT_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
13 PIT1_DBG_DIS_CM7_1	PIT_1 Debug Disable Cortex-M7_1 Specifies whether PIT_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
12 PIT0_DBG_DIS_CM7_1	PIT_0 Debug Disable Cortex-M7_1 Specifies whether PIT_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
11 STM1_DBG_DIS_CM7_1	STM_1 Debug Disable Cortex-M7_1 Specifies whether STM_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
10 STM0_DBG_DIS_CM7_1	STM_0 Debug Disable Cortex-M7_1 Specifies whether STM_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enters Debug mode 1b - Remains functional and unimpacted
9 SWT1_DBG_DIS_CM7_1	SWT_1 Debug Disable Cortex-M7_1 Specifies whether SWT_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
8 SWT0_DBG_DIS_CM7_1	SWT_0 Debug Disable Cortex-M7_1 Specifies whether SWT_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
7 RTC_DBG_DIS_CM7_1	RTC Debug Disable Cortex-M7_1 Specifies whether RTC enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
6 EMIOS2_DBG_DIS_CM7_1	EMIOS2 debug disable bit for CM7_1. Set this bit 1 to disable the debug of IP. 0b - EMIOS2 enters debug mode when CM7_1 enters debug mode. 1b - EMIOS2 remains functional and is not impacted when CM7_1 enters debug mode.
5 EMIOS1_DBG_DIS_CM7_1	EMIOS1 debug disable bit for CM7_1. Set this bit 1 to disable the debug of IP. 0b - EMIOS1 enters debug mode when CM7_1 enters debug mode. 1b - EMIOS1 remains functional and is not impacted when CM7_1 enters debug mode.
4 EMIOS0_DBG_DIS_CM7_1	eMIOS_0 Debug Disable Cortex-M7_1 Specifies whether eMIOS_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
3	LCU_1 Debug Disable Cortex-M7_1

Table continues on the next page...

Table continued from the previous page...

Field	Function
LCU1_DBG_DIS_CM7_1	Specifies whether LCU_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
2 LCU0_DBG_DIS_CM7_1	LCU_0 Debug Disable Cortex-M7_1 Specifies whether LCU_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
1 FCCU_DBG_DIS_CM7_1	FCCU Debug Disable Cortex-M7_1 Specifies whether FCCU enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
0 EDMA_DBG_DIS_CM7_1	eDMA Debug Disable Cortex-M7_1 Specifies whether eDMA enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted

37.2.35 Read Write GPR On Destructive Reset 9 (DCMRWD9)

Offset

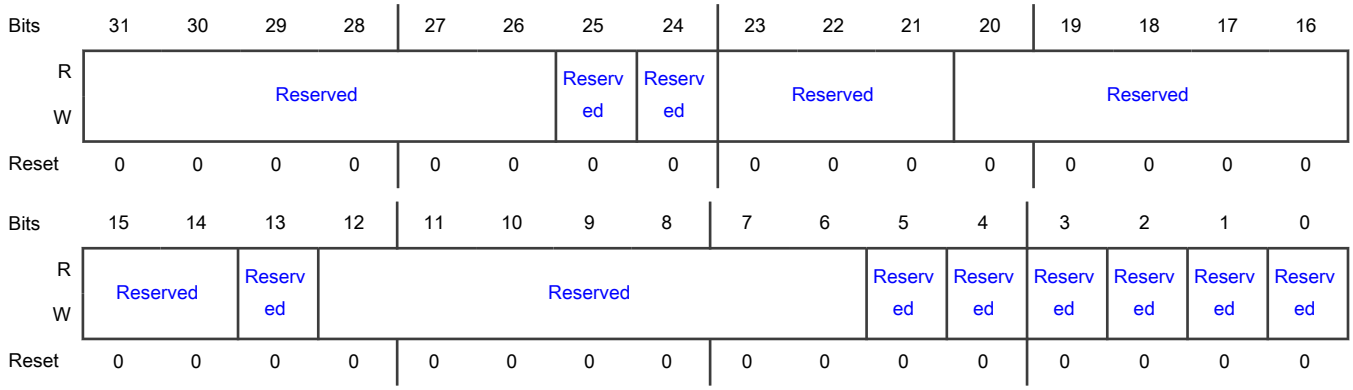
Register	Offset
DCMRWD9	520h

Function

Provides module debug disable information.

This register resets after destructive reset 9.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 —	Reserved
24 —	Reserved
23-21 —	Reserved
20-14 —	Reserved
13 —	Reserved
12-6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 —	Reserved
1 —	Reserved
0 —	Reserved

37.2.36 Read Write GPR On Functional Reset 1 (DCMRWF1)

Offset

Register	Offset
DCMRWF1	600h

Function

Contains information related to:

- Voltage dividers, LFAST clocks, and supply voltage monitoring.
- I/O configurations.

This field resets after functional reset 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EMAC_TX...	Reserved			VDD_1_5...	VDD_HV_...	VDD_HV_...	VSS_LV_...	SUPPLY_MON_SEL			SUPPLY_...	Reserved	Reserved	Reserved	STANDBY...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31	EMAC_TX_RMII_CLK Loopback Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
EMAC_TX_RMII_CLK_LPBCK_EN	Enables the EMAC_TX_RMII_CLK loopback. 0b - Disables 1b - Enables
30-28 —	Reserved
27 VDD_1_5_VLT_DVDR_EN	VDD1P5 Voltage Divider Enable Enables the VDD1P5 2:1 divider for voltage measurement using the supply voltage that ADC monitors. 0b - Disables 1b - Enables
26 VDD_HV_B_VLT_DVDR_EN	VDD_HV_B Voltage Divider Enable Enables the VDD_HV_B 2:1 divider for voltage measurement by using the supply voltage that ADC monitors. 0b - Disables 1b - Enables
25 VDD_HV_A_VLT_DVDR_EN	VDD_HV_A Voltage Divider Enable Enables the VDD_HV_A 2:1 divider for voltage measurement by using supply voltage that ADC monitors. 0b - Disables 1b - Enables
24 VSS_LV_ANMUX_EN	VSS_LV Monitoring Enable Enables VSS_LV monitoring. **NOTE** You must write 1 to this field (with DCMRWF1[SUPPLY_MON_EN] = 1b0 for VSS_LV monitoring by ADC0). 0b - Disables 1b - Enables
23-21 SUPPLY_MON_SEL	Supply Monitoring Select Selects the source of voltage that ADC uses for supply monitoring. **NOTE** • The SUPPLY_MON_SEL configurations are effective only when SUPPLY_MON_EN is 1. • When SUPPLY_MON_EN is 0 and VSS_LV_ANMUX_EN is 1, VSS_LV is monitored. 000b - VDD_HV_A_DIV

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - VDD_HV_B_DIV 010b - VDD_1.5_DIV 011b - VDD_2.5_OSC 100b - VDD1.1_PD1_HOT_POINT 101b - VDD1.1_PD1_COLD_POINT 110b - VDD1.1_PLL 111b - VDD1.1_PD0
20 SUPPLY_MON_EN	Supply Monitoring Enable Enables the ADC supply voltage monitoring. 0b - Disables 1b - Enables
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 STANDBY_IO_CONFIG	Standby I/O Configuration Controls the I/O state in Standby mode: • For standby I/O configuration standby entry, you must write 0 to this field before performing I/O configurations in standby entry sequence. • For standby I/O configuration standby exit, you must write 1 to this field after performing I/O configurations on standby exit. 0b - Standby I/O configuration standby entry 1b - Standby I/O configuration standby exit
15 VDD_HV_B_IO_CTRL_LATCH	VDD_HV_B I/O Control Latch Specifies whether the VDD_HV_B domain pins function as normal or are latched. This field manages I/O control latching in low-frequency Run mode to reduce power consumption on the VDD_HV_B domain pins. **NOTE** This field must remain 0, except in FIRC 3 MHz and FIRC 187.5 kHz operation modes. 0b - Function as normal

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Are latched
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 EMAC_CONF_SEL	Selects between MII and RMII mode of ethernet. 0b - MII mode 1b - RMII mode
6 —	Reserved
5 FCCU_SW_NCF3	FCCU Software NCF 3 Specifies whether NCF 3 to FCCU is generated. For the exact FCCU slot, see the "Fault Collection and Control Unit (FCCU)" chapter. 0b - Not generated 1b - Generated
4 FCCU_SW_NCF2	FCCU Software NCF 2 Specifies whether NCF 2 to FCCU is generated. For the exact FCCU slot, see the "Fault Collection and Control Unit (FCCU)" chapter. 0b - Not generated 1b - Generated

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 FCCU_SW_NCF1	FCCU Software NCF 1 Specifies whether NCF 1 to FCCU is generated. For the exact FCCU slot, see the "Fault Collection and Control Unit (FCCU)" chapter. 0b - Not generated 1b - Generated
2 FCCU_SW_NCF0	FCCU Software NCF 0 Specifies whether NCF0 to FCCU is generated. For the exact FCCU slot, see the "Fault Collection and Control Unit (FCCU)" chapter. 0b - Not generated 1b - Generated
1 CAN_TIMESTAMP_EN	FlexCAN Timestamp Enable Enables the FlexCAN timestamping feature. 0b - Disables 1b - Enables
0 CAN_TIMESTAMP_SEL	FlexCAN Timestamp Select Selects either EMAC or STM for FlexCAN timestamping. 0b - EMAC 1b - STM0

37.2.37 Read Write GPR On Functional Reset 2 (DCMRWF2)

Offset

Register	Offset
DCMRWF2	604h

Function

Contains information related to:

- WKPU source select.
- PGOOD polarity.
- HSE_B gasket bypass.
- Bypass standby exit.

This register resets after functional reset 2.

Diagram

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

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

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23-22 —	Reserved
21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 HSE_GSKT_BYPASS	HSE_B Gasket Bypass Enables the HSE_B IAHB gasket behavior out of Standby mode. • If you write 0 to this field, the DCF client controls the HSE_B IAHB gasket bypass configuration. The system must continue to run on FIRC, and if intended to run on PLL, a functional reset must be asserted. • If you write 1 to this field, the HSE_B IAHB gasket is bypassed out of standby. 0b - Not bypassed 1b - Bypassed
15-12 —	Reserved
11 —	Reserved
10-9 —	Reserved
8 —	Reserved
7 —	Reserved
6 SIRC_TRIM_BYP_STDBY_EXT	SIRC Trim Bypass Standby Exit Controls the bypassing of SIRC trimming on standby exit. 0b - Not bypassed 1b - Bypassed
5	PMC Trim MC_RGM DCF Bypass Standby Exit

Table continues on the next page...

Table continued from the previous page...

Field	Function
PMC_TRIM_RGM_DCF_BYP_STDBY_EXT	Controls the bypassing of PMC trimming and MC_RGM loading on standby exit. 0b - Not bypassed 1b - Bypassed
4 FIRC_TRIM_BYP_STDBY_EXT	FIRC Trim Bypass Standby Exit Controls the bypassing of FIRC trimming on standby exit. 0b - Not bypassed 1b - Bypassed
3 DCM_SCAN_BYP_STDBY_EXT	DCM Scan Bypass Standby Exit Controls the bypassing of DCM scanning on standby exit. 0b - Not bypassed 1b - Bypassed
2 —	Reserved
1 —	Reserved
0 —	Reserved

37.2.38 Read Write GPR On Functional Reset 4 (DCMRWF4)

Offset

Register	Offset
DCMRWF4	60Ch

Function

Contains information related to:

- Mux mode enable.
- Input bypass.

This field resets after functional reset 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved	Reserved	Reserved					Reserved		Reserved	Reserved	CM7_1_C...	CM7_0_C...	GLITCH_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27-23 —	Reserved
22-21 —	Reserved
20 —	Reserved
19 —	Reserved
18 CM7_1_CPUWAIT	Cortex-M7_1 CPU Wait Enables the configuration to place the Cortex-M7_1 core into Wait mode. 0b - Disables CPUWAIT 1b - Enables CPUWAIT
17	Cortex-M7_0 CPU Wait

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_0_CPUWAIT	Enables the configuration to place the Cortex-M7_0 core in CPU Wait mode. 0b - Disables CPUWAIT 1b - Enables CPUWAIT
16 GLITCH_FIL_TRG_IN3_BYP	Glitch Filter TRGMUX Input 3 Bypass Selects whether to bypass or filter out the pulse. If this field = 0, it enables glitch filter on TRGMUX input 60, and if the field = 1, it bypasses glitch filter on TRGMUX input 60. 0b - Enables 1b - Bypasses
15 GLITCH_FIL_TRG_IN2_BYP	Glitch Filter TRGMUX Input 2 Bypass Selects whether to bypass or filter out the pulse. If this field = 0, it enables glitch filter on TRGMUX input 61, and if the field = 1, it bypasses glitch filter on TRGMUX input 61. 0b - Enables 1b - Bypasses
14 GLITCH_FIL_TRG_IN1_BYP	Glitch Filter TRGMUX Input 1 Bypass Selects whether to bypass or filter out the pulse. If this field = 0, it enables glitch filter on TRGMUX input 62, and if the field = 1, it bypasses glitch filter on TRGMUX input 62. 0b - Enables 1b - Bypasses
13 GLITCH_FIL_TRG_IN0_BYP	Glitch Filter TRGMUX Input 0 Bypass Selects whether to bypass or filter out the pulse. If this field = 0, it enables glitch filter on TRGMUX input 63, and if the field = 1, it bypasses glitch filter on TRGMUX input 63. 0b - Enables 1b - Bypasses
12-11 —	Reserved
10 MUX_MODE_EN_ADC2_S9	Mux Mode Enable ADC2 Standard Channel 9 Controls the selection of GPIOs to drive ADC_2 standard channel 9. 0b - GPIO_132 1b - GPIO_46
9 MUX_MODE_EN_ADC2_S8	Mux Mode Enable ADC2 Standard Channel 8 Controls the selection of GPIOs to drive ADC_2 standard channel 8. 0b - GPIO_133 1b - GPIO_45

Table continues on the next page...

Table continued from the previous page...

Field	Function
8-7 —	Reserved
6 MUX_MODE_EN_ADC1_S23	Mux Mode Enable ADC_1 Standard Channel 23 Controls the selection of GPIOs to drive ADC_1 standard channel 23. 0b - GPIO_125 1b - GPIO_146
5 MUX_MODE_EN_ADC1_S22	Mux Mode Enable ADC_1 Standard Channel 22 Controls the selection of GPIOs to drive ADC_1 standard channel 22. 0b - GPIO_124 1b - GPIO_145
4 MUX_MODE_EN_ADC1_S15	Mux Mode Enable ADC_1 Standard Channel 15 Controls the selection of GPIOs to drive ADC_1 standard channel 15. 0b - GPIO_4 1b - GPIO_33
3 MUX_MODE_EN_ADC1_S14	Mux Mode Enable ADC_1 Standard Channel 14 Controls the selection of GPIOs to drive ADC_1 standard channel 14. 0b - GPIO_69 1b - GPIO_32
2 MUX_MODE_EN_ADC0_S9	Mux Mode Enable ADC_0 Standard Channel 9 Controls the selection of GPIOs to drive ADC_0 standard channel 9. 0b - GPIO_1 1b - GPIO_46
1 MUX_MODE_EN_ADC0_S8	Mux Mode Enable ADC_0 Standard Channel 8 Controls the selection of GPIOs to drive ADC_0 standard channel 8. 0b - GPIO_0 1b - GPIO_45
0 —	Reserved

37.2.39 Read Write GPR On Functional Reset 5 (DCMRWF5)

Offset

Register	Offset
DCMRWF5	610h

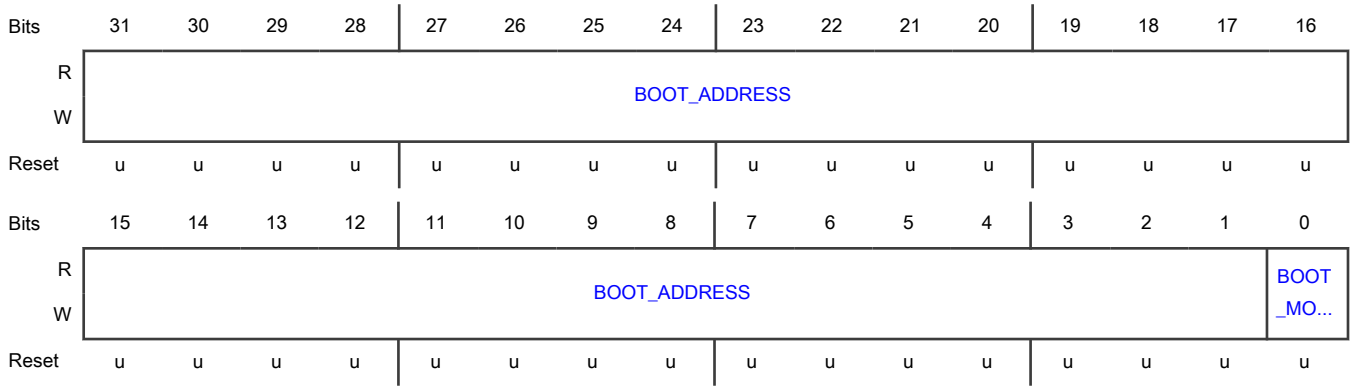
Function

Contains boot address and boot mode.

This register resets after functional reset 5.

The reset value of this register is undefined on reset and is loaded from the flash memory contents at the end of the reset sequence.

Diagram



Fields

Field	Function
31-1 BOOT_ADDRESS	Boot Address Specifies the Cortex-M7_0 base address of the vector table to be used after exiting Standby mode (only to be considered in Fast Standby mode).
0 BOOT_MODE	Boot Mode Selects the type of Boot mode after exiting Standby mode. 0b - Normal 1b - Fast Standby

37.2.40 Read-Only GPR On PMCPOR Reset 1 (DCMROPP1)

Offset

Register	Offset
DCMROPP1	700h

Function

Resets after PMCPOR reset 1 and captures the status of the functional reset sequence process when POR_WDG overflows.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	POR_WDG...	POR_WDG...	POR_WDG...	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	POR_WDG...	Reserved	Reserved	POR_WDG...	Reserved
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	POR_WDG...	Reserved	Reserved	POR_WDG...	POR_WDG...	Reserved	Reserved	Reserved	POR_WDG...	POR_WDG...	POR_WDG...	POR_WDG...	POR_WDG...	POR_WDG...	POR_WDG...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 POR_WDG_STAT31	POR_WDG Status 31 Captures the status of the MC_RGM reset event (if occurred) while the chip is in Standby mode. **NOTE** This field is used only for standby sequence monitoring. 0b - Not detected 1b - Detected
30 POR_WDG_STAT30	POR_WDG Status 30 Captures the status of standby exit acknowledgement by MC_PCU when POR_WDG overflows. **NOTE** This field is used only for standby sequence monitoring. 0b - Not acknowledged

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Acknowledged
29 POR_WDG_STAT29	POR_WDG Status 29 Captures the status of the MC_ME standby entry request that MC_ME initiates when POR_WDG overflows. **NOTE** This field is used only for standby sequence monitoring. 0b - Active 1b - Inactive
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 POR_WDG_STAT20	POR_WDG Status 20 Specifies the status of the functional reset sequence process, DEST0, when POR_WDG overflows. 0b - Inactive 1b - Active
19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 —	Reserved
17 POR_WDG_STAT17	POR_WDG Status 17 Specifies the status of the functional reset sequence process, FUNC10, when POR_WDG overflows. 0b - Inactive 1b - Active
16 —	Reserved
15 —	Reserved
14 POR_WDG_STAT14	POR_WDG Status 14 Specifies the status of the functional reset sequence process, FUNC9, when POR_WDG overflows. 0b - Inactive 1b - Active
13 —	Reserved
12 —	Reserved
11 POR_WDG_STAT11	POR_WDG Status 11 Specifies the status of the functional reset sequence process, FUNC8, when POR_WDG overflows. 0b - Inactive 1b - Active
10 POR_WDG_STAT10	POR_WDG Status 10 Specifies the status of the functional reset sequence process, FUNC7, when POR_WDG overflows. 0b - Inactive 1b - Active
9 —	Reserved
8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved
6 POR_WDG_STAT6	POR_WDG Status 6 Specifies the status of the functional reset sequence process, FUNC6, when POR_WDG overflows. 0b - Inactive 1b - Active
5 POR_WDG_STAT5	POR_WDG Status 5 Specifies the status of the functional reset sequence process, FUNC5, when POR_WDG overflows. 0b - Inactive 1b - Active
4 POR_WDG_STAT4	POR_WDG Status 4 Specifies the status of the functional reset sequence process, FUNC4, when POR_WDG overflows. 0b - Inactive 1b - Active
3 POR_WDG_STAT3	POR_WDG Status 3 Specifies the status of the functional reset sequence process, FUNC3, when POR_WDG overflows. 0b - Inactive 1b - Active
2 POR_WDG_STAT2	POR_WDG Status 2 Specifies the status of the functional reset sequence process, FUNC2, when POR_WDG overflows. 0b - Inactive 1b - Active
1 POR_WDG_STAT1	POR_WDG Status 1 Specifies the status of the functional reset sequence process, FUNC1, when POR_WDG overflows. 0b - Inactive 1b - Active
0 POR_WDG_STAT0	POR_WDG Status 0 Specifies the status of the functional reset sequence process, FUNC0, when POR_WDG overflows. 0b - Inactive 1b - Active

37.2.41 Read-Only GPR On PMCPOR Reset 2 (DCMROPP2)

Offset

Register	Offset
DCMROPP2	704h

Function

Resets after PMCPOR reset 2 and captures the MC_RGM functional or external event status when POR_WDG overflows.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	POR_WDG...	POR_WDG...	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	POR_WDG...	Reserved	Reserved	Reserved	POR_WDG...
W		W1C	W1C									W1C				W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved	POR_WDG...	Reserved	Reserved	POR_WDG...	POR_WDG...	POR_WDG...	POR_WDG...	Reserved	POR_WDG...	POR_WDG...	Reserved	Reserved	POR_WDG...
W				W1C			W1C	W1C	W1C	W1C		W1C	W1C			W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 POR_WDG_STAT62	POR_WDG Status 62 Specifies the value of MC_RGM.FES[DEBUG_FUNC] when POR_WDG overflows. 0b - 0 1b - 1
29 POR_WDG_STAT61	POR_WDG Status 61 Specifies the value of MC_RGM.FES[SW_FUNC] when POR_WDG overflows. 0b - 0 1b - 1
28	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 POR_WDG_STAT52	POR_WDG Status 52 Specifies the value of MC_RGM.FES[HSE_BOOT_RST] when POR_WDG overflows. 0b - 0 1b - 1
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 POR_WDG_STAT48	POR_WDG Status 48 Specifies the value of MC_RGM.FES[HSE_SWT_RST] when POR_WDG overflows. 0b - 0 1b - 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 POR_WDG_STAT44	POR_WDG Status 44 Specifies the value of MC_RGM.FES[PLL_AUX] when POR_WDG overflows. 0b - 0 1b - 1
11 —	Reserved
10 —	Reserved
9 POR_WDG_STAT41	POR_WDG Status 41 Specifies the value of MC_RGM.FES[JTAG_RST] when POR_WDG overflows. 0b - 0 1b - 1
8 POR_WDG_STAT40	POR_WDG Status 40 Specifies the value of MC_RGM.FES[Reserved] when POR_WDG overflows. 0b - 0 1b - 1
7 POR_WDG_STAT39	POR_WDG Status 39 Specifies the value of MC_RGM.FES[SWT1_RST] when POR_WDG overflows. 0b - 0 1b - 1
6 POR_WDG_STAT38	POR_WDG Status 38 Specifies the value of MC_RGM.FES[SWT0_RST] when POR_WDG overflows. 0b - 0 1b - 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 —	Reserved
4 POR_WDG_STAT36	POR_WDG Status 36 Specifies the value of MC_RGM.FES[ST_DONE] when POR_WDG overflows. 0b - 0 1b - 1
3 POR_WDG_STAT35	POR_WDG Status 35 Specifies the value of MC_RGM.FES[FCCU_RST] when POR_WDG overflows. 0b - 0 1b - 1
2 —	Reserved
1 —	Reserved
0 POR_WDG_STAT32	POR_WDG Status 32 Specifies the value of MC_RGM.FES[F_EXR] when POR_WDG overflows. 0b - 0 1b - 1

37.2.42 Read-Only GPR On PMCPOR Reset 3 (DCMROPP3)

Offset

Register	Offset
DCMROPP3	708h

Function

Resets after PMCPOR reset 3 and captures the MC_RGM destructive event status when POR_WDG overflows.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	POR_WDG...	POR_WDG...	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	POR_WDG...	POR_WDG...	POR_WDG...
W		W1C	W1C											W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POR_WDG...	POR_WDG...	Reserved	POR_WDG...	Reserved	POR_WDG...	POR_WDG...	POR_WDG...	Reserved	POR_WDG...	Reserved	POR_WDG...	POR_WDG...	Reserved	Reserved	POR_WDG...
W	W1C	W1C		W1C		W1C	W1C	W1C		W1C		W1C	W1C			W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 POR_WDG_STAT94	POR_WDG Status 94 Specifies the value of MC_RGM.DES[DEBUG_DEST] when POR_WDG overflows. 0b - 0 1b - 1
29 POR_WDG_STAT93	POR_WDG Status 93 Specifies the value of MC_RGM.DES[SW_DEST] when POR_WDG overflows. 0b - 0 1b - 1
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 POR_WDG_STAT82	POR_WDG Status 82 Specifies the value of MC_RGM.DES[HSE_SNVS_RST] when POR_WDG overflows. 0b - 0 1b - 1
17 POR_WDG_STAT81	POR_WDG Status 81 Specifies the value of MC_RGM.DES[HSE_TMPR_RST] when POR_WDG overflows. 0b - 0 1b - 1
16 POR_WDG_STAT80	POR_WDG Status 80 Specifies the value of MC_RGM.DES[CM7_CORE_CLK_FAIL] when POR_WDG overflows. 0b - 0 1b - 1
15 POR_WDG_STAT79	POR_WDG Status 79 Specifies the value of MC_RGM.DES[SYS_DIV_FAIL] when POR_WDG overflows. 0b - 0 1b - 1
14 POR_WDG_STAT78	POR_WDG Status 78 Specifies the value of MC_RGM.DES[HSE_CLK_FAIL] when POR_WDG overflows. 0b - 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - 1
13 —	Reserved
12 POR_WDG_STAT76	POR_WDG Status 76 Specifies the value of MC_RGM.DES[AIPS_PLAT_CLK_FAIL] when POR_WDG overflows. 0b - 0 1b - 1
11 —	Reserved
10 POR_WDG_STAT74	POR_WDG Status 74 Specifies the value of MC_RGM.DES[CORE_CLK_FAIL] when POR_WDG overflows. 0b - 0 1b - 1
9 POR_WDG_STAT73	POR_WDG Status 73 Specifies the value of MC_RGM.DES[PLL_LOL] when POR_WDG overflows. 0b - 0 1b - 1
8 POR_WDG_STAT72	POR_WDG Status 72 Specifies the value of MC_RGM.DES[SWT2_RST] when POR_WDG overflows. 0b - 0 1b - 1
7 —	Reserved
6 POR_WDG_STAT70	POR_WDG Status 70 Specifies the value of MC_RGM.DES[MC_RGM_FRE] when POR_WDG overflows. 0b - 0 1b - 1
5 —	Reserved
4	POR_WDG Status 68

Table continues on the next page...

Table continued from the previous page...

Field	Function
POR_WDG_STAT68	Specifies the value of MC_RGM.DES[STCU_URF] when POR_WDG overflows. 0b - 0 1b - 1
3 POR_WDG_STAT67	POR_WDG Status 67 Specifies the value of MC_RGM.DES[FCCU_FTR] when POR_WDG overflows. 0b - 0 1b - 1
2 —	Reserved
1 —	Reserved
0 POR_WDG_STAT64	POR_WDG Status 64 Specifies the value of MC_RGM.DES[F_POR] when POR_WDG overflows. 0b - 0 1b - 1

37.2.43 Read-Only GPR On PMCPOR Reset 4 (DCMROPP4)

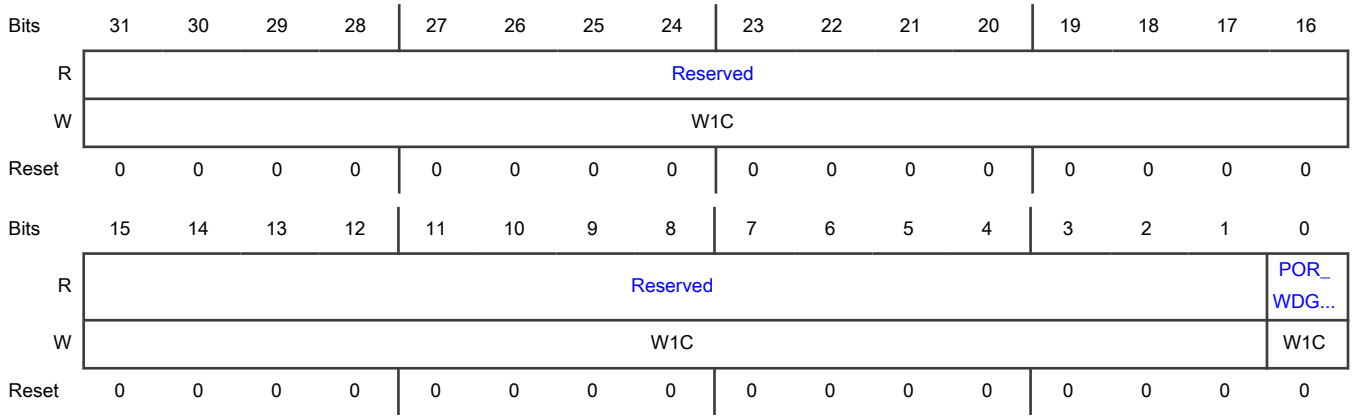
Offset

Register	Offset
DCMROPP4	70Ch

Function

Resets after PMCPOR reset 4 and captures the POR_WDG reset event if POR_WDG initiates a POR sequence.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 POR_WDG_STAT96	POR_WDG Status 96 Specifies the status of POR_WDG. If this field = 1, it indicates that a stuck scenario is detected and the chip POR event is raised. See POR_WDG_STAT[95:0] for the chip status when POR_WDG overflows. 0b - Inactive 1b - Active

Chapter 38
Device Configuration Module (DCM)

38.1 Introduction

38.1.1 Overview

DCM controls:

- LC
- DCF client population
- Debug authorization (Export Control mode)

The module also establishes a RoT for the chip by parsing the master root key and other security records.

This figure is a pictorical representation of DCM:

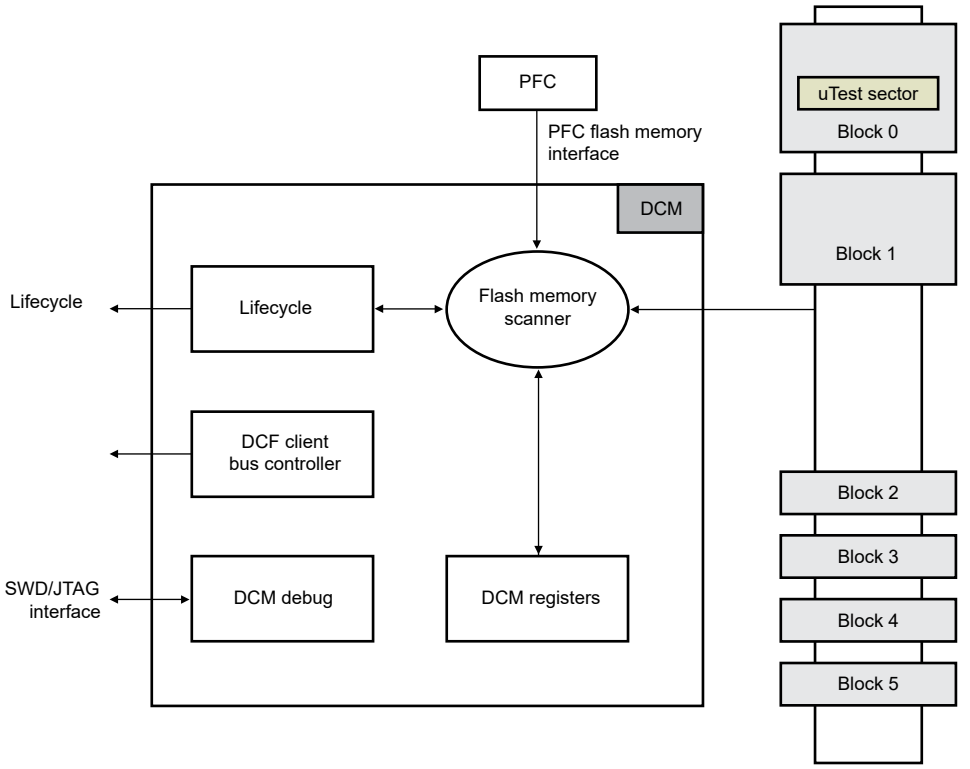


Figure 143. DCM block diagram

38.1.2 Features

- Scans flash memory and configures the system for:
  - LC detection
  - DCF records configuration using flash memory
- Allows debug features for flash memory content and the chip

- Provides debug enable control
- Provides a valid boot address detection
- Enables the DCF client to be writable via the IPS bus
- Parameterizes control to mask a set of DCF client chip select
- Supports temporary advancement of the LC
- Controls chip debug enablement

38.1.3 Modes of operation

DCM operates identically in all system modes of operation.

38.2 Functional description

DCM provides information about the current state and configuration of the system that you could use to:

- Configure the application software.
- Debug the system.

38.2.1 DCF mechanism

The DCF mechanism handles chip parameter settings via the OTP flash memory.

You can store a series of DCF records in flash memory, and each record is 64 bits in length. The chip processes these records during the system reset sequence before the CPU leaves reset.

A DCF record contains:

- A start record. Placed at UTEST_OFFSET, it indicates to the chip that the specified data records must be processed. See Table 192 that shows a general format of the start record.
- Application-specific number of data records. These follow the start record, as required.
- A stop record. Its presence indicates that the processing must stop. See Table 194 for a general format of the stop record. Only the STOP field needs to be 1 to form a stop record—all other fields are ignored. Also, an unprogrammed location in Utest flash memory is interpreted as a stop record.

Table 192. DCF start record

05AA 55AFh
0000 0000h

Table 193. DCF stop record

Reserved	1
Reserved	

A data record is structured in a way similar to a CPU-write instruction—comprising a 15-bit client select field, a 15-bit address field, and a 32-bit payload data field plus a STOP field. See Table 194 for details.

In a data record, the value of the STOP field must always be 0. Some clients support a parity (PRTY) field too.

Table 194. Format of a DCF data record

WDATA[31:0]			
CS[14:0]	ADDR[16:2]	PRTY	STOP

If *n* data records are to be stored in Utest, the data structure must be as shown in the next table.

Table 195. Series of DCF records in Utest

Address offset	Data			
00h	05AA 55AFh			
04h	0000 0000h			STOP = 0
08h	WDATA[31:0]			
0Ch	CS[14:0]	ADDR[16:2]	PRTY	STOP = 0
10h	WDATA[31:0]			
14h	CS[14:0]	ADDR[16:2]	PRTY	STOP = 0
8n - 1 + 0h	Reserved			
8n - 1 + 4h	Reserved			1
8n + 0h				
8n + 4h				

An unprogrammed record must not exist in the data structure because that is interpreted as a stop record. This leads to the subsequent records being ignored. In this case:

- You need to program the records in several sessions, each time appending new records at the end of the list, as shown in the next figure.
- A record may overwrite a client's value defined by a previous record. However, not all clients allow overwrites— that depends on their individiual implementation.

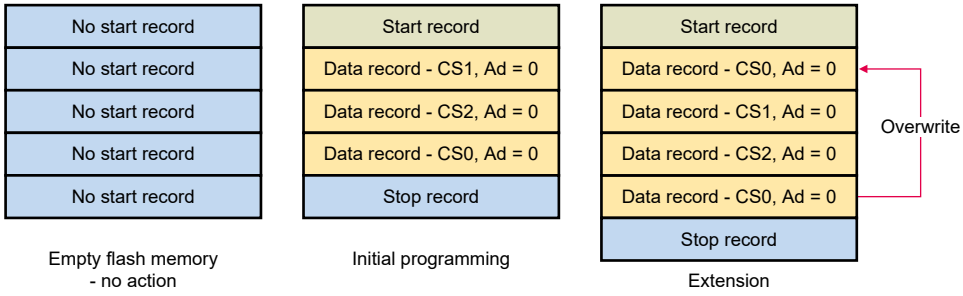


Figure 144. Appending DCF records

Some of the DCF records require parity bits, and the parity scheme used is even parity. So, the number of 1s in the WDATA and PRTY fields needs to be even. For example, if the WDATA field has the value 0000_0001h, the value of the PRTY field needs to be 1 so that the total number of 1s is even.

NOTE

You must ensure that DCF records are programmed uninterruptedly and the process completes without an error.

38.2.2 DCF error recording

DCF errors are recorded in DCM for both types of clients—spread spectrum safe and normal. See DCF client error mechanism for details on spread spectrum safe clients.

38.2.3 DCF client error mechanism

The DCM consists of DCMMISC[DCMCERS] for detecting faulty DCF records. The DCMSRRn registers capture the details up to 16 faulty records. The CERS bit can be cleared by writing 1 to it.

While scanning the flash, in case if a faulty record is encountered, the DCMMISC[DCMCERS] gets set indicating that there is atleast one faulty record. In such a case, the user can identify the details about the faulty record in the DCMSRRn registers and should update the flash memory with a new record, provided the record is not write-once.

For example, in the image below, if the faulty DCF record is DCF2, then the correct DCF record must also be DCF2. Even in case if the faulty record is updated, the DCM stores the faulty record via the DCMMISC[CERS] and DCMSRRn registers.

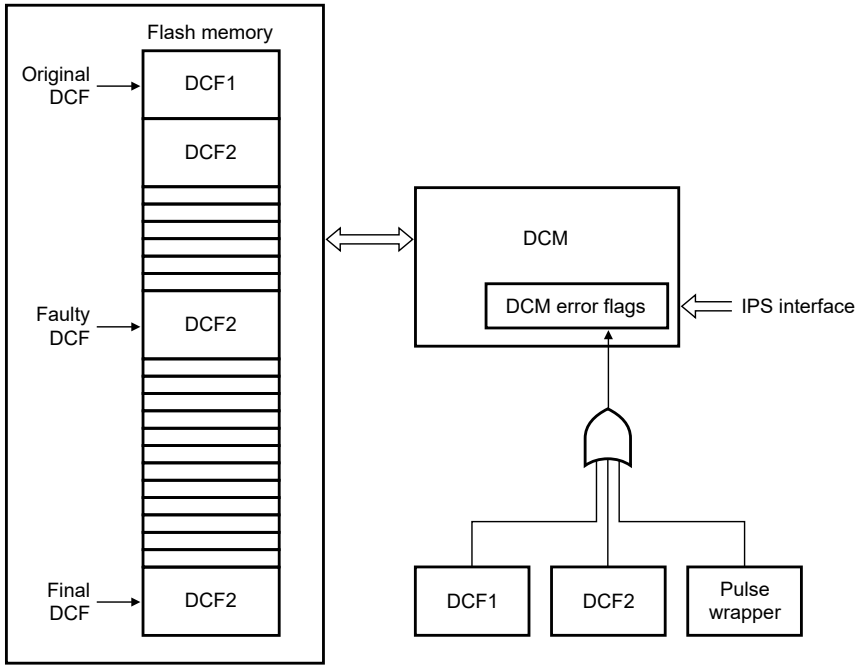


Figure 145. DCF error and CERS handling

38.2.4 DCF error detection mechanism

This figure shows that a faulty DCF record is loaded between the original and final DCF record sets.

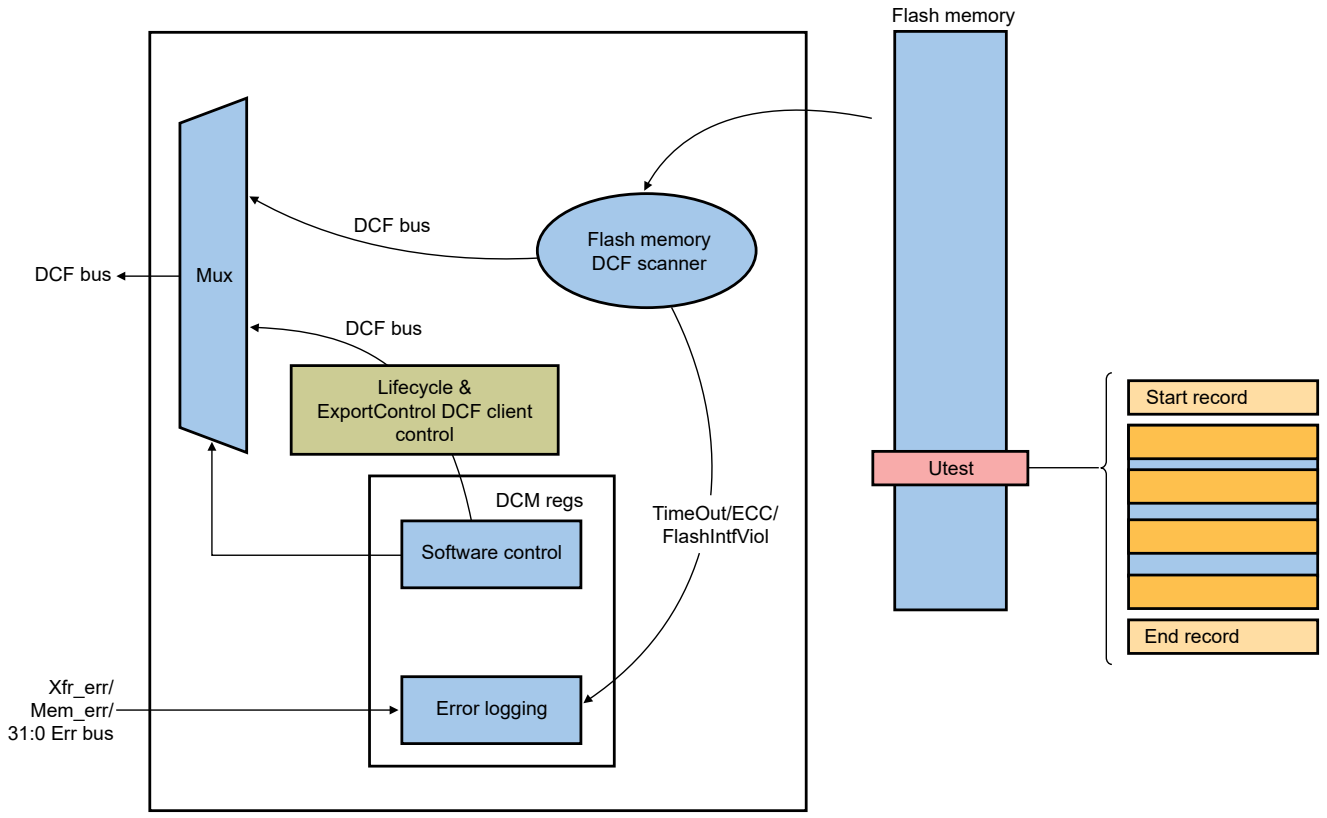


Figure 146. DCF error detection mechanism

38.2.5 LC

DCM determines the LC of the chip by reading the LC slots from the Utest flash memory. This read operation is performed during the reset phase, with normal timings. The operating monitors and an ECC check protect the operation. Additionally, a set of sanity checks executed over the LC read data guarantee the integrity of the final LC value.

At the end of the reset phase, the LC contains one of the following values:

- OEM production (OEM_PROD)
- In field (IN_FIELD)
- Pre-FA
- FA

The DCM LC progresses in the direction shown in this figure:

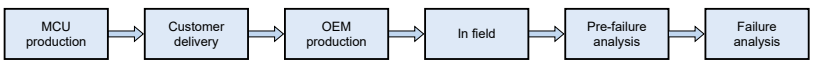


Figure 147. LC sequence

The LC is written into six slots, 128 bits each, and at fixed locations in the Utest flash memory blockblock. Each LC slot is read in a single atomic operation and is organized in two types of fields:

- Valid
- Invalid

Depending on the possible combinations of data programmed into these fields, each LC slot indicates one of the four possible statuses as shown in Table 196. To know more about LC slots, see Table 197.

Table 196. LC slot status

LC slots		LC slot value
Valid field (64 bits)	Invalid field (64 bits)	
Erased	Erased	Erased
Marked	Erased	Active
Marked	Marked	Inactive
Other values		Illegal

In this case, "Marked" refers to a value that is configured based on the bit pattern 55AA_50AF_55AA_50AFh, and "Erased" is detected using the bit pattern FFFF_FFFF_FFFF_FFFFh.

Table 197. LC slots

LC slot 0 MCU_PROD	LC slot 1 CUST_DEL	LC slot 2 OEM_PROD	LC slot 3 IN_FIELD	LC slot 4 Pre-FA	LC slot 5 FA	Resulting LC
Active	Erased	Erased	Erased	Erased	Erased	MCU_PROD
Inactive	Active	Erased	Erased	Erased	Erased	CUST_DEL
Inactive	Inactive	Active	Erased	Erased	Erased	OEM_PROD
Inactive	Inactive	Inactive	Active	Erased	Erased	IN_FIELD
Inactive	Inactive	Inactive	Inactive	Active	Erased	PFA
Inactive or Erased	Inactive	Inactive	Inactive	Inactive	Active	FA
Erased	Erased	Erased	Erased	Erased	Erased	System reset
Illegal[1]	Illegal	Illegal	Illegal	Erased	Erased	IN_FIELD

1. Values other than the ones specified in this table for Erased, Active, and Inactive are treated as illegal.

NOTE

When triggering DCM for rescanning the software, you must ensure that the flash memory program and erase fields are not set. Otherwise, DCM does not configure the chip and ignores all data returned from the flash memory. This results in LC becoming IN_FIELD. All DCF clients indicate their default values in this case.

38.3 DCM register descriptions

38.3.1 DCM memory map

DCM base address: 402A_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	DCM Status (DCMSTAT)	32	R	See section
4h	LC and LC Control (DCMLCC)	32	RW	See section
8h	LC Scan Status (DCMLCS)	32	RW	0000_0000h
1Ch	DCM Miscellaneous (DCMMISC)	32	RW	0000_0001h
20h	Debug Status and Configuration (DCMDEB)	32	RW	0000_0000h
2Ch	DCF Error Count (DCMEC)	32	R	0000_0000h
30h	DCF Scan Report (DCMSRR1)	32	RW	0000_0000h
34h	DCF Scan Report (DCMSRR2)	32	RW	0000_0000h
38h	DCF Scan Report (DCMSRR3)	32	RW	0000_0000h
3Ch	DCF Scan Report (DCMSRR4)	32	RW	0000_0000h
40h	DCF Scan Report (DCMSRR5)	32	RW	0000_0000h
44h	DCF Scan Report (DCMSRR6)	32	RW	0000_0000h
48h	DCF Scan Report (DCMSRR7)	32	RW	0000_0000h
4Ch	DCF Scan Report (DCMSRR8)	32	RW	0000_0000h
50h	DCF Scan Report (DCMSRR9)	32	RW	0000_0000h
54h	DCF Scan Report (DCMSRR10)	32	RW	0000_0000h
58h	DCF Scan Report (DCMSRR11)	32	RW	0000_0000h
5Ch	DCF Scan Report (DCMSRR12)	32	RW	0000_0000h
60h	DCF Scan Report (DCMSRR13)	32	RW	0000_0000h
64h	DCF Scan Report (DCMSRR14)	32	RW	0000_0000h
68h	DCF Scan Report (DCMSRR15)	32	RW	0000_0000h
6Ch	DCF Scan Report (DCMSRR16)	32	RW	0000_0000h
80h	LC Scan Status 2 (DCMLCS_2)	32	RW	0000_0000h

38.3.2 DCM Status (DCMSTAT)

Offset

Register	Offset
DCMSTAT	0h

Function

Indicates the status of DCM at different stages.

NOTE

This register resets on functional reset.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0	DCMDBGPS	DCMOTAS	DCMUTS	0			DCMLCST	0		DCMERR	DCMDONE
W																
Reset	0	0	0	0	0	0	u[1]	1	0	1	1	1	0	0	0	1

1. The value of this field is loaded from flash memory based on chip configuration.

Fields

Field	Function
31-19 —	Reserved
18-16 —	Reserved
15-12 —	Reserved
11 —	Reserved
10 DCMDBGPS	Debug Password Scanning Status Indicates the DCM debug password scanning status. 0b - Completed with errors 1b - Completed successfully
9 DCMOTAS	DCM OTA Scanning Status (valid only when the value of the DCMDONE field is 1) 0b - Completed with errors 1b - Completed successfully
8 DCMUTS	DCM Utest DCF Scanning Status (valid only if DCMDONE bit is set)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This bit always returns 0 in In Field LC. 0b - DCM Utest DCF completed with errors. 1b - DCM Utest DCF completed successfully.
7-5 —	Reserved
4 DCMLCST	LC Scanning Status Indicates the DCM LC scanning status. This field is valid only if the value of the DCMDONE field is 1. Also, it always returns 0 in the IN_FIELD phase of the LC. For details, see LC. 0b - Completed with errors 1b - Completed successfully
3-2 —	Reserved
1 DCMERR	DCM completion with error status (valid only if DCMDONE bit is set) 0b - DCM completed with success. 1b - DCM completed with error.
0 DCMDONE	DCM Scanning Status Indicates whether the DCM scanning is in progress or complete. 0b - Running 1b - Completed

38.3.3 LC and LC Control (DCMLCC)

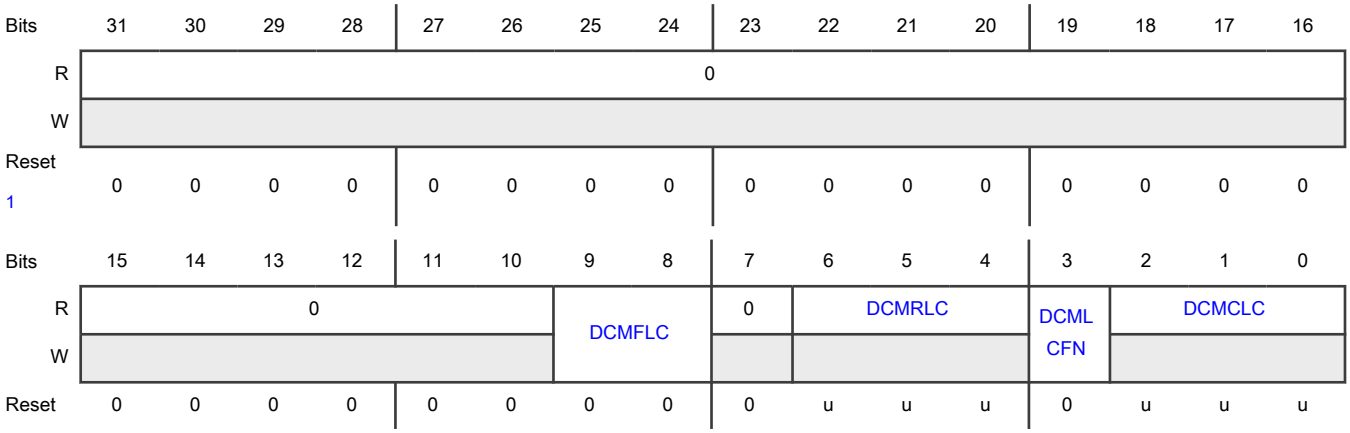
Offset

Register	Offset
DCMLCC	4h

Function
Resets on the functional reset.

NOTE
The DCMLCFN and DCMFLC fields of this register reset on destructive reset.

Diagram



1. Post reset, the reset value of this register is 0000_0077h, and after scanning, it changes according to the programmed value.

Fields

Field	Function
31-10 —	Reserved
9-8 DCMFLC	Force Next LC Provides provisions for a temporary LC update. LC can be temporarily forced as follows: • MCU_PROD => Force one => CUST_DEL (CD) • CUST_DEL => Force one => OEM_PROD • CUST_DEL => Force two => IN_FIELD • OEM_PROD => Force one => IN_FIELD • IN_FIELD => Force not allowed to move LC • Pre-FA => Force not allowed to move LC • FA => Force not allowed to move LC 00b - No force 01b - Force LC to one next 10b - Force LC to two next (only possible in CD) 11b - Reserved (no effect)
7 —	Reserved
6-4 DCMRLC	Real LC Projects the real LC of the chip.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The LC can move in this sequence: (110 : MCU_PROD) > (011 : CUST_DEL) > (010 : OEM_PROD) > (111 : IN_FIELD) > (001 : Pre-FA) > (000 : FA) 000b - FA 001b - Pre-FA 010b - OEM_PROD 011b - CUST_DEL 100b - Reserved 101b - Reserved 110b - MCU_PROD 111b - IN_FIELD
3 DCMLCFN	Force LC Indicates if the LC is normal or forced on the current LC. If the value of this field is 1, the value configured according to the DCMLCF register is forced on the current LC whereas the real LC remains the same. 0b - Normal 1b - Forced on current LC
2-0 DCMCLC	Current LC Projects the current LC of the chip. The LC can move in this sequence: (110 : MCU_PROD) > (011 : CUST_DEL) > (010 : OEM_PROD) > (111 : IN_FIELD) > (001 : Pre-FA) > (000 : FA) 000b - FA 001b - Pre-FA 010b - OEM_PROD 011b - CUST_DEL 100b - Reserved 101b - Reserved 110b - MCU_PROD 111b - IN_FIELD

38.3.4 LC Scan Status (DCMLCS)

Offset

Register	Offset
DCMLCS	8h

Function

Stores the status of LC scanning. By default, the status of each LC is "not yet scanned."

This register:

- Resets on destructive reset.
- Always returns 0 in a valid IN_FIELD LC (in LC without an error).

This register captures the errors related to LC scanning on each of these resets: POR, destructive, and functional. If an error is captured, its status in this register is cleared by writing 1 to the corresponding field or to any of the destructive or POR events.

All LC slot errors are captured and cleared independently.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMLCFE5	DCMLCE5	DCMLCC5			DCMLCSS5	DCMLCFE4	DCMLCE4	DCMLCC4			DCMLCSS4	DCMLCFE3	DCMLCE3
W			W1C	W1C	W1C			W1C	W1C	W1C	W1C			W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMLCC3			DCMLCSS3	DCMLCFE2	DCMLCE2	DCMLCC2			DCMLCSS2	DCMLCFE1	DCMLCE1	DCMLCC1			DCMLCSS1
W	W1C			W1C	W1C	W1C	W1C			W1C	W1C	W1C	W1C			W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMLCFE5	Pre-FA Flash Memory Error Check Indicates the status of the Pre-FA flash memory error check. 0b - Successful 1b - Failed
28 DCMLCE5	Pre-FA ECC Errors Indicates if Pre-FA is successful or has ECC errors. 0b - No errors 1b - Marking error
27-25 DCMLCC5	Pre-FA Marking Status Indicates the Pre-FA marking status. These errors may cause this field to indicate the "not scanned yet" status:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If the reading completes too early and DCM has not yet scanned the LC. • If there is an error in the flash memory after completion of the reading. 000b - Not scanned yet 001b - Marked as active 010b - Marked as inactive 011b - Region is erased/virgin 101b - Marked as inactive by an unknown pattern 110b - Scanning timed out
24 DCMLCSS5	Pre-FA Scan Status Indicates the status of the Pre-FA scan. 0b - Successful 1b - Errors exist
23 DCMLCFE4	IN_FIELD Flash Memory Error Check Indicates the status of IN_FIELD flash memory error check. 0b - Successful 1b - Failed
22 DCMLCE4	IN_FIELD ECC Errors Indicates if IN_FIELD has ECC errors. 0b - No errors 1b - Errors exist
21-19 DCMLCC4	IN_FIELD Marking Status Indicates the IN_FIELD marking status. These errors may cause this field to indicate the "not scanned yet" status: • If the reading completes too early and DCM has not yet scanned the LC. • If there is an error in the flash memory after completion of the reading. 000b - Not scanned yet 001b - Marked as active 010b - Marked as inactive 011b - Region is erased/virgin 101b - Marked as inactive by an unknown pattern 110b - Scanning timed out
18	IN_FIELD Scan Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
DCMLCSS4	Indicates the status of the IN_FIELD scan. 0b - No errors 1b - Errors exist
17 DCMLCFE3	OEM_PROD Flash Memory Error Check Indicates the status of the OEM_PROD flash memory error check. 0b - Successful 1b - Failed
16 DCMLCE3	OEM_PROD ECC Errors Indicates if OEM_PROD has ECC errors. 0b - No errors 1b - Errors exist
15-13 DCMLCC3	OEM_PROD Marking Indicates the OEM_PROD marking status. These errors may cause this field to indicate the "not scanned yet" status: • If the reading completes too early and DCM has not yet scanned the LC. • If there is an error in the flash memory after completion of the reading. 000b - Not scanned yet 001b - Marked as active 010b - Marked as inactive 011b - Region is erased/virgin 101b - Marked as inactive by an unknown pattern 110b - Scanning timed out
12 DCMLCSS3	OEM_PROD Scan Status Indicates the status of OEM_PROD scan. 0b - No errors 1b - Errors exist
11 DCMLCFE2	CUST_DEL Flash Memory Error Check Indicates the status of CUST_DEL flash memory error check. 0b - Successful 1b - Failed
10 DCMLCE2	ECC Errors In CUST_DEL Indicates if CUST_DEL has ECC errors.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No errors 1b - Errors exist
9-7 DCMLCC2	CUST_DEL Marking Indicates the CUST_DEL marking status. These errors may cause this field to indicate the "not scanned yet" status: • If the reading completes too early and DCM has not yet scanned the LC. • If there is an error in the flash memory after completion of the reading. 000b - Not scanned yet 001b - Marked as active 010b - Marked as inactive 011b - Region is erased/virgin 101b - Marked as inactive by an unknown pattern 110b - Scanning timed out
6 DCMLCSS2	CUST_DEL Scan Status Indicates the status of the CUST_DEL scan. 0b - No errors 1b - Errors exist
5 DCMLCFE1	MCU_PROD Flash Memory Error Check Indicates the status of flash memory error check. 0b - Successful 1b - Failed
4 DCMLCE1	MCU_PROD ECC Errors Indicates if MCU_PROD has ECC errors. 0b - No errors 1b - Errors exist
3-1 DCMLCC1	MCU_PROD Marking Indicates the MCU_PROD marking status. These errors may cause this field to indicate the "not scanned yet" status: • If the reading completes too early and DCM has not yet scanned the LC. • If there is an error in the flash memory after completion of the reading. 000b - Not scanned yet

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - Marked as active 010b - Marked as inactive 011b - Region is erased/virgin 101b - Marked as inactive by an unknown pattern 110b - Scanning timed out
0 DCMLCSS1	MCU_PROD Scan Status Indicates the MCU_PROD scan status. 0b - No errors 1b - Errors exist

38.3.5 DCM Miscellaneous (DCMMISC)

Offset

Register	Offset
DCMMISC	1Ch

Function

Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	DCMCERS	0				0							
W				W1C												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		0		DCMDBGE	DCMDBGT	0									
W					W1C	W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
30 —	Reserved
29 —	Reserved
28 DCMCERS	DCF Client Errors Records the status of errors from DCF clients. 0b - No errors on any of the DCF clients 1b - Atleast one safety DCF client has an error
27-24 —	Reserved
23-14 —	Reserved
13-12 —	Reserved
11 DCMDBGE	DCM ECC error on DBG sections This bit is set if there is any ECC error during scanning of NXP_PWD, CUST_PWD, or UID. 0b - No error on DBG section 1b - DBG section error
10 DCMDBGT	DBG Section Error Indicates if there is a DCM flash memory timeout error in DBG sections. The value of this field is 1 in case a timeout error occurs when scanning NXP_PWD, CUST_PWD, or UID. 0b - No error 1b - Error exists
9-0 —	Reserved

38.3.6 Debug Status and Configuration (DCMDEB)

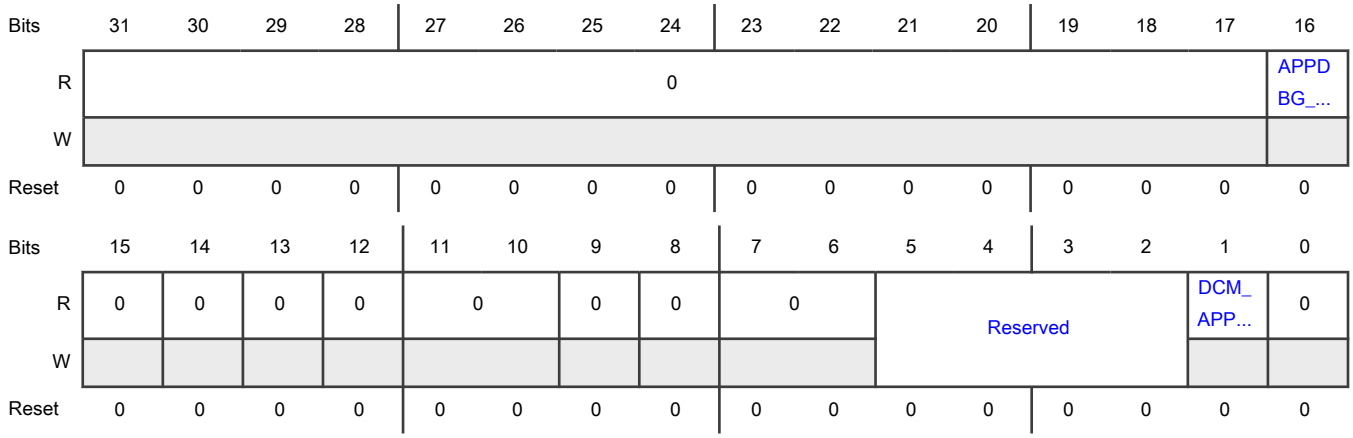
Offset

Register	Offset
DCMDEB	20h

Function

Resets on destructive reset.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 APPDBG_STAT_SOC	Application Debug Status Indicates the application debug status of the chip. 0b - Disabled 1b - Enabled
15 —	Reserved
14 —	Reserved
13 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 —	Reserved
11-10 —	Reserved
9 —	Reserved
8 —	Reserved
7-6 —	Reserved
5-2 —	Reserved
1 DCM_APPDBG_STAT	DCM Authentication Engine Status Indicates the DCM authentication engine status for the application core. **NOTE** This value of this field is 0 in Non-Export Control mode. 0b - Disabled 1b - Enabled
0 —	Reserved

38.3.7 DCF Error Count (DCMEC)

Offset

Register	Offset
DCMEC	2Ch

Function

Indicates the display count for the number of errors encountered when scanning the flash memory during DCF scanning.

This register resets on destructive reset.

Diagram

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

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

Fields

Field	Function
31-16 —	Reserved
15-0 DCMECT	Error Count Indicates the display count for the number of errors encountered when scanning the flash memory during DCF scanning.

38.3.8 DCF Scan Report (DCMSRR1)

Offset

Register	Offset
DCMSRR1	30h

Function

Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMDCFT1	DCMESD1	DCMESF1	DCMDCFF1			0				DCMDCFE1			
W			W1C	W1C	W1C	W1C							W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMDCFE1															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT1	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD1	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF1	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF1	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE1	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

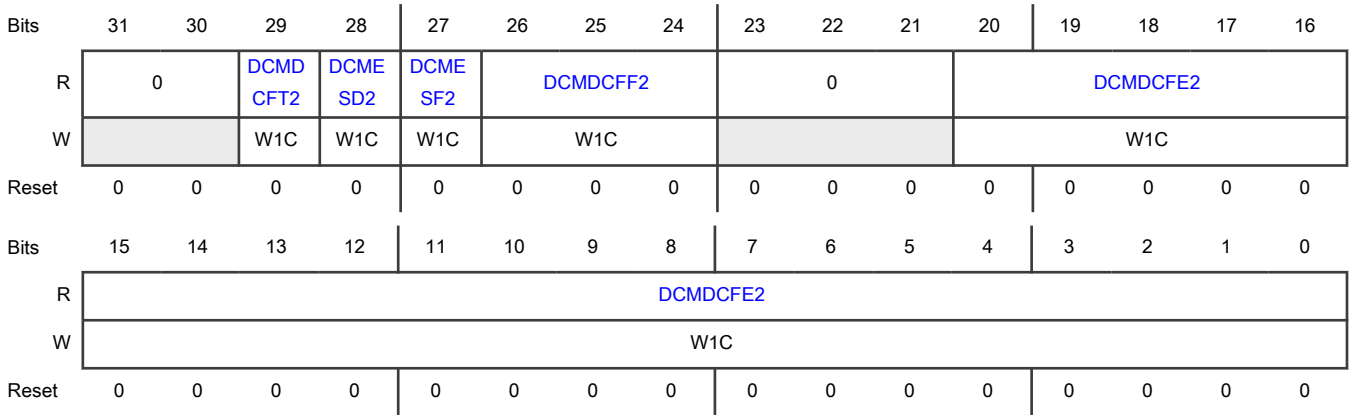
38.3.9 DCF Scan Report (DCMSRR2)

Offset

Register	Offset
DCMSRR2	34h

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT2	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD2	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF2	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF2	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE2	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

38.3.10 DCF Scan Report (DCMSRR3)

Offset

Register	Offset
DCMSRR3	38h

Function

Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMDCFT3	DCMESD3	DCMESF3	DCMDCFF3			0				DCMDCFE3			
W			W1C	W1C	W1C	W1C							W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMDCFE3															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT3	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD3	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF3	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF3	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE3	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

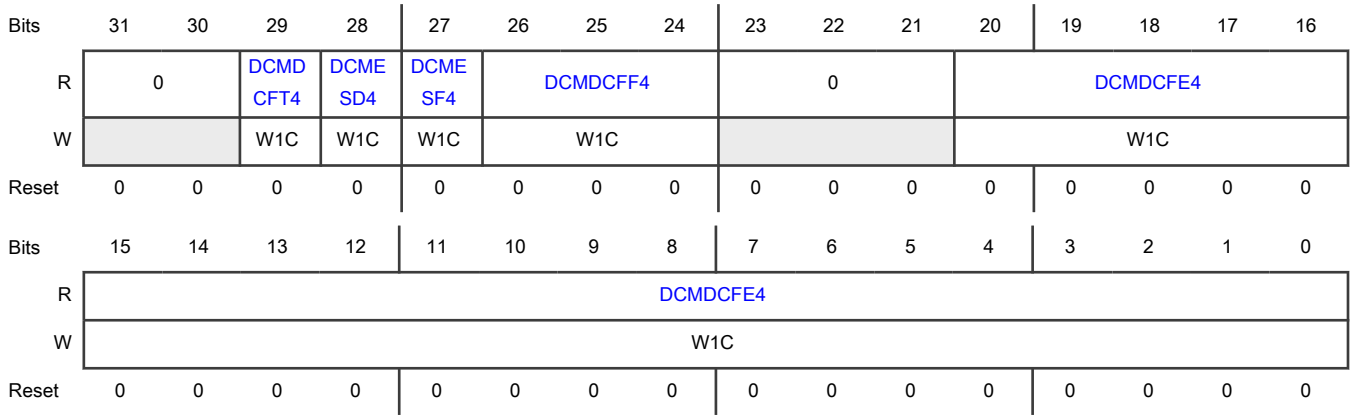
38.3.11 DCF Scan Report (DCMSRR4)

Offset

Register	Offset
DCMSRR4	3Ch

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT4	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD4	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF4	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF4	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE4	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

38.3.12 DCF Scan Report (DCMSRR5)

Offset

Register	Offset
DCMSRR5	40h

Function

Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMDCFT5	DCMESD5	DCMESF5	DCMDCFF5			0				DCMDCFE5			
W			W1C	W1C	W1C	W1C							W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMDCFE5															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT5	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD5	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF5	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF5	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE5	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

38.3.13 DCF Scan Report (DCMSRR6)

Offset

Register	Offset
DCMSRR6	44h

Function
Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMDCFT6	DCMESD6	DCMESF6	DCMDCFF6			0				DCMDCFE6			
W			W1C	W1C	W1C	W1C							W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMDCFE6															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT6	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD6	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF6	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF6	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE6	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

38.3.14 DCF Scan Report (DCMSRR7)

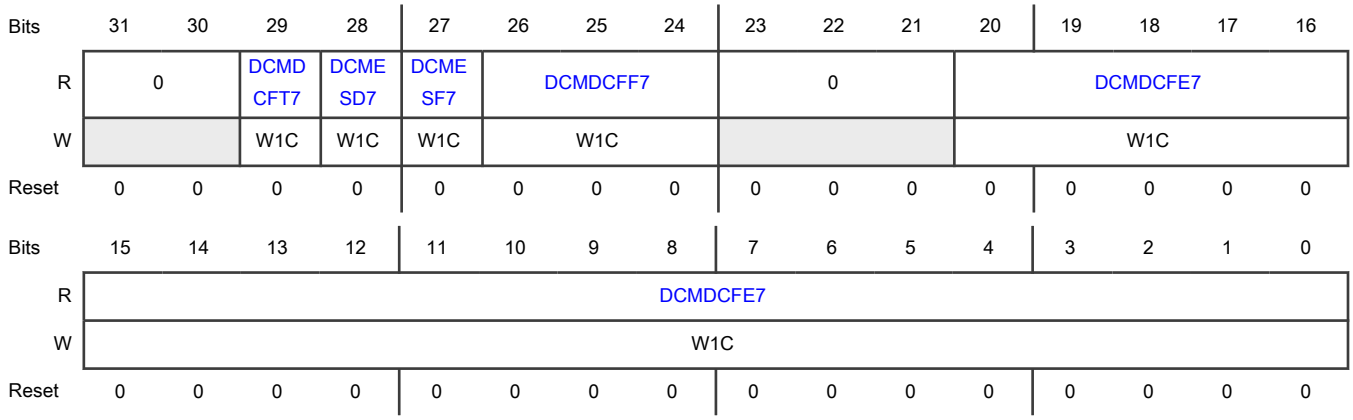
Offset

Register	Offset
DCMSRR7	48h

Function

Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT7	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD7	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF7	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF7	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE7	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

38.3.15 DCF Scan Report (DCMSRR8)

Offset

Register	Offset
DCMSRR8	4Ch

Function
Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMDCFT8	DCMESD8	DCMESF8	DCMDCFF8			0				DCMDCFE8			
W			W1C	W1C	W1C	W1C							W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMDCFE8															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT8	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD8	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF8	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF8	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE8	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

38.3.16 DCF Scan Report (DCMSRR9)

Offset

Register	Offset
DCMSRR9	50h

Function

Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMDCFT9	DCMESD9	DCMESF9	DCMDCFF9			0				DCMDCFE9			
W			W1C	W1C	W1C	W1C							W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMDCFE9															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT9	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD9	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF9	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF9	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE9	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

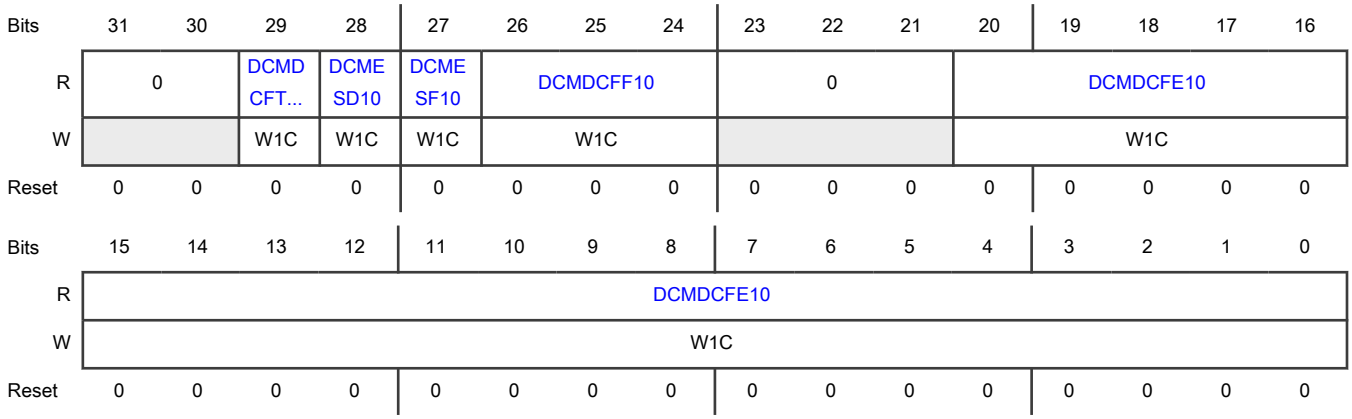
38.3.17 DCF Scan Report (DCMSRR10)

Offset

Register	Offset
DCMSRR10	54h

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT10	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD10	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF10	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF10	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE10	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

38.3.18 DCF Scan Report (DCMSRR11)

Offset

Register	Offset
DCMSRR11	58h

Function

Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMDCFT...	DCMESD11	DCMESF11	DCMDCFF11			0				DCMDCFE11			
W			W1C	W1C	W1C	W1C							W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMDCFE11															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT11	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD11	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF11	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF11	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE11	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

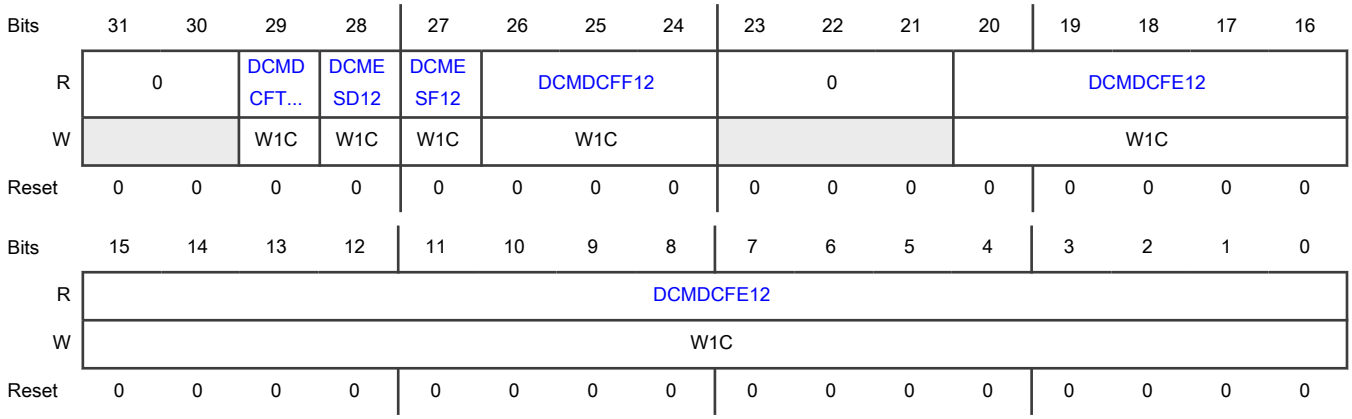
38.3.19 DCF Scan Report (DCMSRR12)

Offset

Register	Offset
DCMSRR12	5Ch

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT12	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD12	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF12	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF12	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE12	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

38.3.20 DCF Scan Report (DCMSRR13)

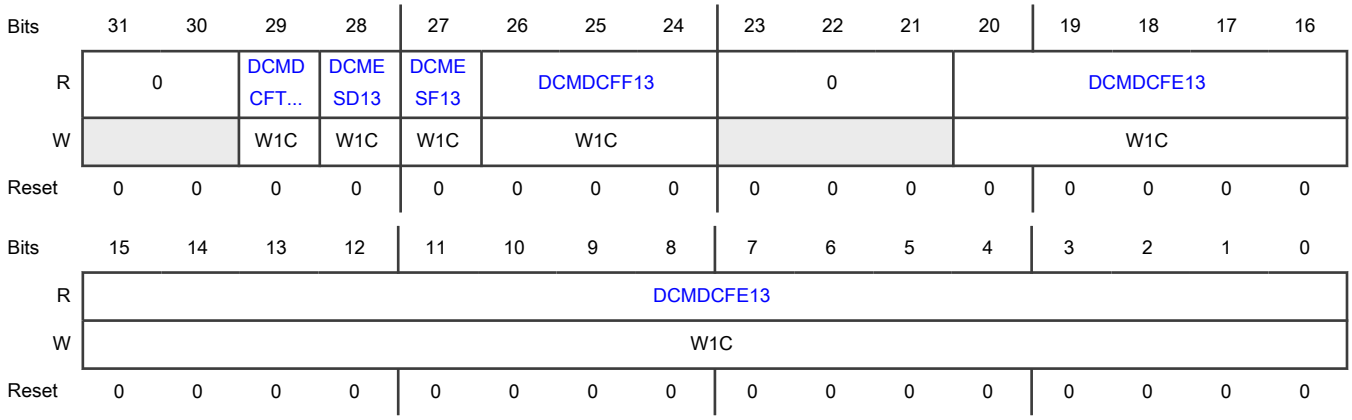
Offset

Register	Offset
DCMSRR13	60h

Function

Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT13	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD13	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF13	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF13	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE13	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

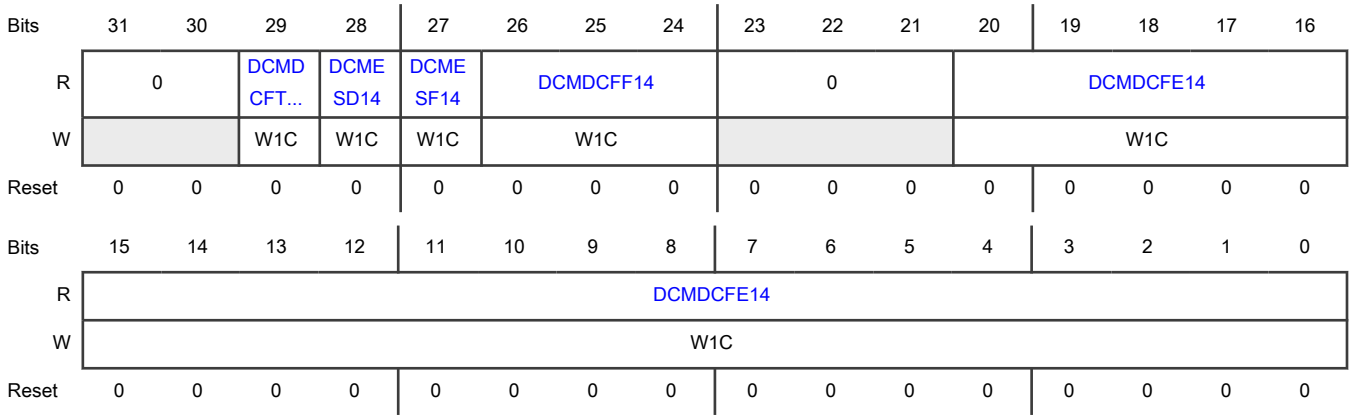
38.3.21 DCF Scan Report (DCMSRR14)

Offset

Register	Offset
DCMSRR14	64h

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT14	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD14	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF14	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF14	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE14	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

38.3.22 DCF Scan Report (DCMSRR15)

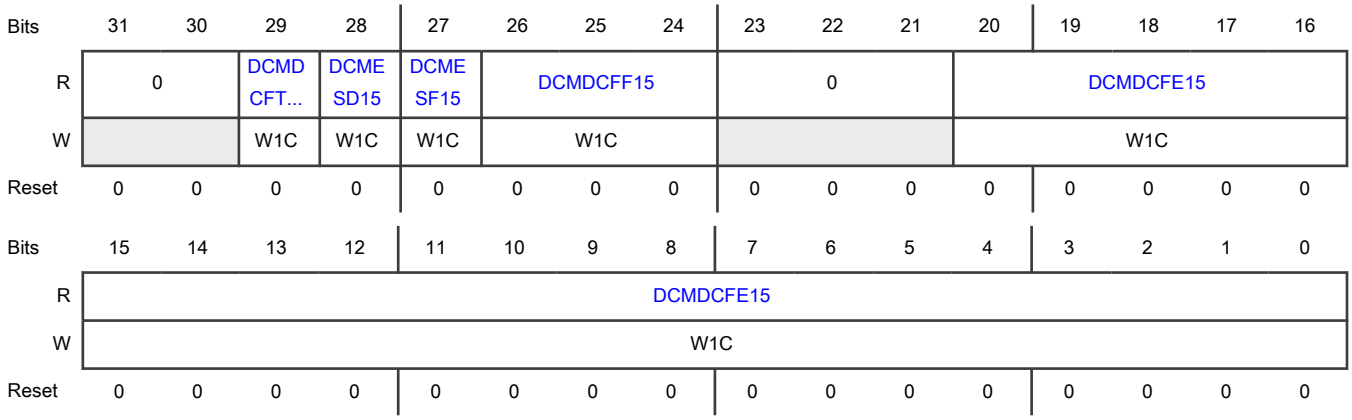
Offset

Register	Offset
DCMSRR15	68h

Function

Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT15	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD15	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF15	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF15	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE15	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

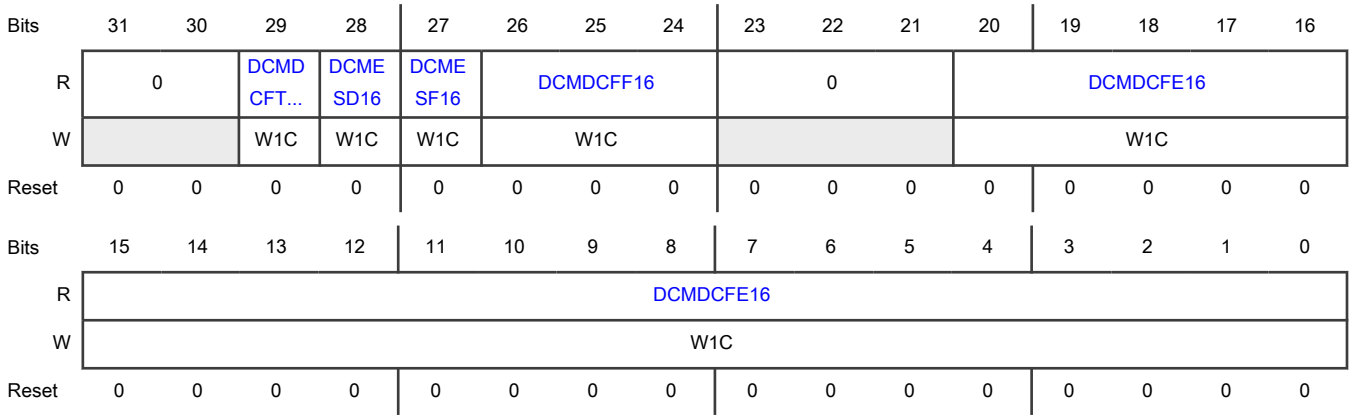
38.3.23 DCF Scan Report (DCMSRR16)

Offset

Register	Offset
DCMSRR16	6Ch

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT16	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD16	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF16	Flash Memory Error Indicates if a flash memory error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF16	DCF Record Location Indicates the DCF record location in flash memory. 010b - Utest region 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE16	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

38.3.24 LC Scan Status 2 (DCMLCS_2)

Offset

Register	Offset
DCMLCS_2	80h

Function

Stores the status of LC scan. The default status of each LC phase is "not yet scanned."

This register resets on destructive reset, and always returns 0 in the IN_FIELD phase of the LC.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										DCMLCFE6	DCMLCE6	DCMLCC6			DCMLCSS6
W											W1C	W1C	W1C			W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5 DCMLCFE6	Flash Memory Error Check Indicates the status of flash memory check. 0b - Successful 1b - Failed

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 DCMLCE6	FA ECC Errors Indicates if ECC errors exist in FA. 0b - No errors 1b - Errors exist
3-1 DCMLCC6	FA Marking Indicates the FA marking status. These errors may cause this field to indicate the "not scanned yet" status: • If the reading completes too early and DCM has not yet scanned the LC. • If there is an error in the flash memory after completion of the reading. 000b - Not scanned yet 001b - Marked as active 010b - Marked as inactive 011b - Region is erased/virgin 101b - Marked as inactive by an unknown pattern 110b - Scanning timed out
0 DCMLCSS6	FA Scan Status Indicates if errors exist in the FA scan. 0b - No errors 1b - Errors exist

38.4 Glossary

FA	Failure analysis
LC	Life cycle
Pre-FA	Pre-failure analysis
RoT	Root of trust

Chapter 39
Messaging Unit (MU)

39.1 Chip-specific MU information

39.1.1 MU instances and configuration

This chip includes MUs for communication across the different cores. Each MU includes two interfaces, MUA and MUB. Two different processors control them for communication.

Table 198 indicates the MUs and their interfaces present in different parts of the MWCT chip family. The table also summarizes the implementation of this module in each chip of the MWCT product series.

NOTE

The HSE_B core controls the MUA interface of MU_0 and MU_1. Therefore, the application core cannot control the interface.

Table 198. MU instances

Instance	MWCT2D17S	MWCT2D16S	MWCT2016S	MWCT2015S	Use case
MU_0	Yes				Communication between HSE_B and application cores
MU_1	Yes				
MU_2	Yes	Yes	No		Communication between application cores

The base address of MU_1's MUB interface is different across the MWCT2xxxS product series because AIPS_2 is unavailable in MWCT2016S and MWCT2015S. Table 199 summarizes this difference.

Table 199. Base-address difference in MU_1's MUB interface across MWCT2xxxS product series

	MWCT2D17S	MWCT2D16S	MWCT2016S	MWCT2015S
Base address	404E_C000h		4039_0000h	

NOTE

For MWCT2D16S, reset value of MUA_VER and MUB_VER register is 0309_000Fh.

39.2 Overview

The Messaging Unit module (MU) enables two processors within the SoC to communicate and coordinate by passing messages (e.g. data, status and control) through the MU interface. The MU also provides the ability for one processor to signal the other processor using interrupts.

Because the MU manages the messaging between processors potentially using different clocks, the MU must synchronize the accesses from one side to the other. The MU accomplishes synchronization using two sets of matching registers (Processor A-facing, Processor B-facing).

39.2.1 Block Diagram

The following is the MU Block Diagram:

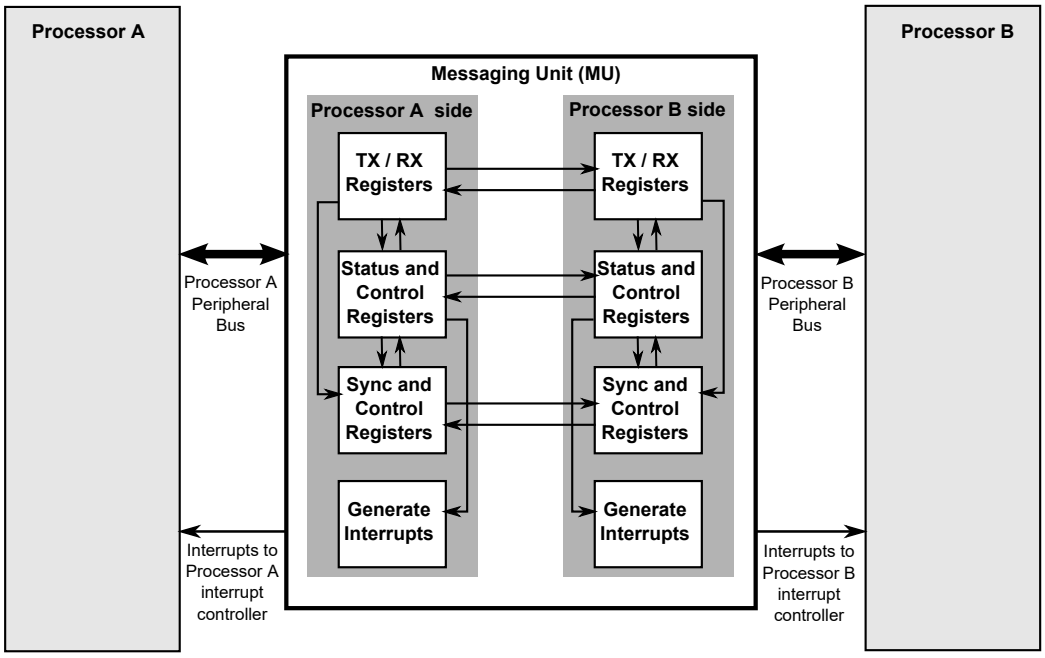


Figure 148. MU Block Diagram

39.2.2 Features

The MU includes the following features:

- Memory-Mapped Registers
  - The MU is connected as a peripheral under the Peripheral bus on Processor A-side and Processor B-side.
- Synchronized Message Transfers between Cores
  - For sending data or messages from one MU-side to the other MU-side, the MUA provides 4 transmit registers and 4 receive registers, the MUB provides 4 transmit registers and 4 receive registers.
  - The transfer of data messages between cores uses transmit empty and receive full flags provided on both sides of the MU.
  - The update of these transmit and receive flags is accomplished using a synchronization mechanism. There is inherent latency between updating the flag on one side and reflecting its status on other side.
  - The MU has a 3 bit Flag Data Register, which can be used to send flag data between two MU sides.
- Interprocessor Interrupts
  - The MU has 9 interrupt sources on each side (Processor A-side, Processor B-side) that are used for signaling the other processor. The interrupts can be used for notification of RX/TX events and general-purpose signaling between the processors. There are 1 general-purpose interrupt requests available and 8 RX/TX interrupt sources.
- MU Reset
  - Each Processor can issue a reset to the entire MU, using a control bit (MUR) in each Processor's Control Register (CR).
  - The MUR bit is a self-clearing bit.

39.3 Functional Description

The Messaging Unit (MU) enables two cores (Processor A or Processor B) to communicate with each other, by passing message/data information to each other, and by enabling one core to wake up the other core using interrupts.

The messaging, control, and status registers of the Processor A and Processor B-side for the MU are mapped to the Processor A and Processor B memory as a regular peripheral. The Peripheral data bus is 32 bits wide inside the MU module.

The messaging logic is used in conjunction with external memory. There are various messaging methods that can be used to implement a messaging protocol. Some of these messages could mean “A message of N words has been written starting at offset X in the memory,” or "The previous data block that was sent has been read." Having the messaging logic independent from the memory array is not restricted to a predefined hardware protocol. On the other hand, the software needed to manage the messaging is short and straightforward.

Most of the messaging mechanisms are symmetric. They are duplicated and are available on both the Processor A-side and the Processor B-side. The messaging mechanisms are:

- 4 32-bit transmit registers, which are each reflected in 4 read-only receive registers in the other processor’s side. You can use these registers to transfer 32-bit word messages or frame information of messages written to the shared memory (number of words, initial address, and message type code).
- A write to a transmit register on the transmitter side clears a “transmitter empty” bit in the Status Register on the transmitter side, and sets a “receiver full” bit in the Status Register on the receiver side. The setting of the bit at the receiver side can optionally trigger an interrupt at the receiver side (maskable receive interrupt).
- A read of one of the receive registers at the receiver side clears the “receiver full” bit in the Status Register at the receiver side, and sets the “transmitter empty” bit in the Status Register on the transmitter side. The setting of the “transmitter empty” bit can optionally trigger an interrupt at the transmitter side (maskable transmit interrupt).
- 1 general purpose interrupt request flags are reflected in the General Status Register(GSR) on the receiver side.
- 3-bit flag data is transmitted from FCR register to xFSR register with the SR[FUP] indicator bit.

A write to a transmit register signals the receiver side that data is ready for retrieval.

- Writing to the transmit register again without verifying that the data was retrieved is prohibited, because the transmitter side has no way of knowing the exact time that the receiver attempts to retrieve the data.
- Before attempting to write the transmit register again, the transmitter side should wait for a “Transmitter Empty” interrupt, or should poll the “Transmitter Empty” bit in the Transmit Status Register.

A read of a receive register signals the transmitter side that data can be written to that register. In the same way, the receiver processor should not read a receive register before receiving a “Receiver Full” interrupt or polling the “Receiver Full” bit in the Receive Status Register.

- Reading the receive register again without verifying that the data was written is prohibited, because the receiver side has no way of knowing the exact time that the transmitter attempts to write the data.
- Before attempting to read the receive register again, the receiver side should wait for a “Receiver Full” interrupt, or should poll the “Receiver Full” bit in the Receive Status Register.

39.3.1 Event Update Timing

Each processor’s MU messaging side (Processor A or Processor B) has a hardware mechanism to send “event update requests” to the other processor’s side. An “event” is considered when any information change should be reflected at the Status Register of the receiving processor. The event update latency is the delay between the event being ready at one processor and the resulting update at the Status Register of the other processor.

- The minimum event latency is “1 clock of the sending side” + “2 1/2 clocks of the receiving side”. The minimum case is if there is no event pending when the new event occurs.
- The maximum event latency is “6 clocks of the sending side” + “6 1/2 clocks of the receiving side.” The maximum case is if the event occurred just after a previous event was sent to the other side. The event update latency varies between the above-mentioned minimum and maximum latencies, depending on the time at which the subsequent event is triggered.

39.3.2 Interrupts

The MU controls one processor interrupt requests to the other processor. This section describes all the interrupts that the module generates.

Below interrupt sources can be generated individually from the MU to the Processors.

- 4 receive interrupts (asserted when the Processors receive full bits are set and enabled in the xRCR register) for each of the receive registers
- 4 transmit interrupts (asserted when the Processor transmit empty bits are set and enabled in the xTCR register) for each of the transmit registers
- 1 general purpose interrupts (asserted when the GIP bits are set and enabled in the xGIER register)
- Core Reset Assertion interrupts(asserted when Processor goes into of Reset state and enabled in RAIE register bit)
- MU Software/Hardware Reset assertion(asserted when MUR and HR bit are written and enabled in MURIE and HRIE register bit)

All the interrupts are maskable in the Processor Control Register (xTCR, xRCR, xGIER, xCR and xCIER). The MU does not assume any internal priority of these interrupts. Multiple interrupts (for example, Receive 0 and Receive 1 interrupts or any of the transmit and general purpose interrupts) can be asserted at one time. The priority of these interrupts should be resolved by the interrupt controller at the chip level.

Triggering any enabled interrupt wakes up the Processor before servicing the interrupt.

The General Purpose Interrupt Pending GIPn bits should be cleared by the software (as part of the interrupt service routine) to de-assert the request to the interrupt controller.

When a Processor writes to the General Interrupt bit (GIRn), the write event is synchronized to the other Processor to set the general interrupt request pending bit (GIPn). When the GIPn bit is set, and if the general purpose interrupt is enabled on the receiving Processor side (GIEn bit is set), then the transmitting Processor general purpose interrupt is issued to the receiving Processor. The receiving Processor clears this interrupt by writing a “1” on the GIPn bit. The interrupt is de-asserted as soon as the GIPn bit is written. The write event of the GIPn bit is synchronized to the other Processor. The synchronized signal clears the GIR bit. The software should not write the GIRn bit again until the GIRn bit is cleared.

Before setting a GIRn bit, you must verify that the GIRn bit is cleared, which means that a general interrupt is not pending. Generally, setting the GIRn bit while the bit has been set to “1” is ignored, but in some cases it may issue a second interrupt.

39.3.3 Resets

The MU has below reset sources, and each reset has a different function from the MU or system perspective.

- Asynchronous system reset:
  - The asynchronous system reset on one MU-side returns all registers on this MU side to their default state.
  - When the asynchronous system reset on any MU-side is asserted, the MURS bit in CR register is set to "1" until the asynchronous system reset ends. Check MURS bit is cleared before starting any access to MU.
- MU software reset:
  - Writing the CR[MUR] bit to "1" causes most of control and status registers to return to their default values and all internal states to be cleared.
  - The instruction immediately following assertion of the MUR bit should not write to MU registers. Such a write may be overwritten by the reset sequence and the register remains with the reset value. You should monitor the value of the SR[MURS] bit to know when the reset sequence on both processors has ended. After the reset sequence on both processors has ended, you can attemp a write to MU registers.

NOTE

MUR bit assertion is a delicate operation because it affects the other side's registers asynchronously. MUR bit assertion may cause unpredictable behavior if, for example, the other Processor is concurrently testing an MU register bit (TE bit in the other Processor TSR register). Before asserting the MUR bit, you should verify that the other Processor is not presently engaged in an MU signalling activity.

39.4 External Signals

There are no Messaging Unit signals directly available externally.

39.5 Application Information

The following are some messaging examples:

- **Passing short messages:** Transmit register(s) can be used to pass short messages from one to 4 words in length for Processor A and 4 for Processor B. For example, when a four-word message is desired, only one of the registers needs to have its corresponding interrupt enable bit set at the receiver side; the message's first three words are written to the registers whose interrupt is masked, and the fourth word is written to the other register (which triggers an interrupt at the receiver side).
- **Passing frame information:** Transmit registers can be used to pass frame information for long messages written to the shared system memory. Such frame information normally includes a start address, number of words, and perhaps a message type code.
- **Passing event notices and requests:** Events and requests that do not include data words can be signaled from the Processor B to the Processor A using the general interrupts, such as acknowledging that a long message was read from the shared system memory.
- **Passing fixed length data:** Formatted data with a fixed length can be written in predetermined locations in the shared memory. A processor can use a general interrupt to signal the other processor that the data is ready.
- **Passing announcements:** The 3 flags can be used by a processor to announce its current program state or other billboard messages to the other processor.

39.5.1 Messaging Protocols using Interrupts

The example below describes a four-word messaging sequence sent by the Processor to the other Processor.

In this example, the first, second, and third receive interrupts are disabled, and the fourth receive interrupt is enabled. We write registers sequentially for n = 0, 1, 2, 3. For n = 0, 1, 2, the interrupts are disabled, therefore no interrupt goes to the other core (although interrupt conditions occur). For n = 3, the interrupt is enabled, and the last Receive Interrupt request is generated.

1. Write Sequence
   - The Processor writes the message information sequentially to its Transmit Registers 0, 1, 2.
   - When the write to the Transmit Register 3 occurs, the RF3 bit of the xRSR is set after synchronization, and it immediately trigger the Receive Full 3 interrupt to the other Processor.
2. Read Sequence
   - The other Processor receives the Receive Full 3 interrupt and starts reading the message transferred from the receive registers.
   - After Receive Register 3 is read, the interrupt bit is cleared.

The following table and diagram describe the messaging model using transmit/receive registers and interrupt messaging protocol.

Table 200. Interrupt Messaging Protocol (Generalized)

Sequence	Action	Description
1	Processor A Data write	A data write to the TRn register by Processor A is immediately reflected in the Processor B RRn register.
2	Clear Tx Empty bit and Set Rx Full bit	The data write to the TRn register • Clears the transmitter empty bit (TEn) in the Processor A Transmit Status Register • Sets the receiver full bit (RFn) in the Processor B Receive Status Register
3	Generate Receive Interrupt request	The setting of the receiver full bit (RFn) in the Receive Status Register generates a Receive Interrupt request to Processor B.
4	Processor B Data read	After receiving the Receive Interrupt request, Processor B performs a data read of the RRn register.
5	Clear Rx Full bit and Set Tx Empty bit	Reading the data out of the RRn register • Clears the receiver full bit (RFn) in the Processor B Receive Status Register • Sets the transmitter empty bit (TEn) in the Processor A Transmit Status Register
6	Generate Transmit Interrupt request	The setting of the transmitter empty bit (TEn) in the Transmit Status Register generates a Transmit Interrupt request to Processor A.

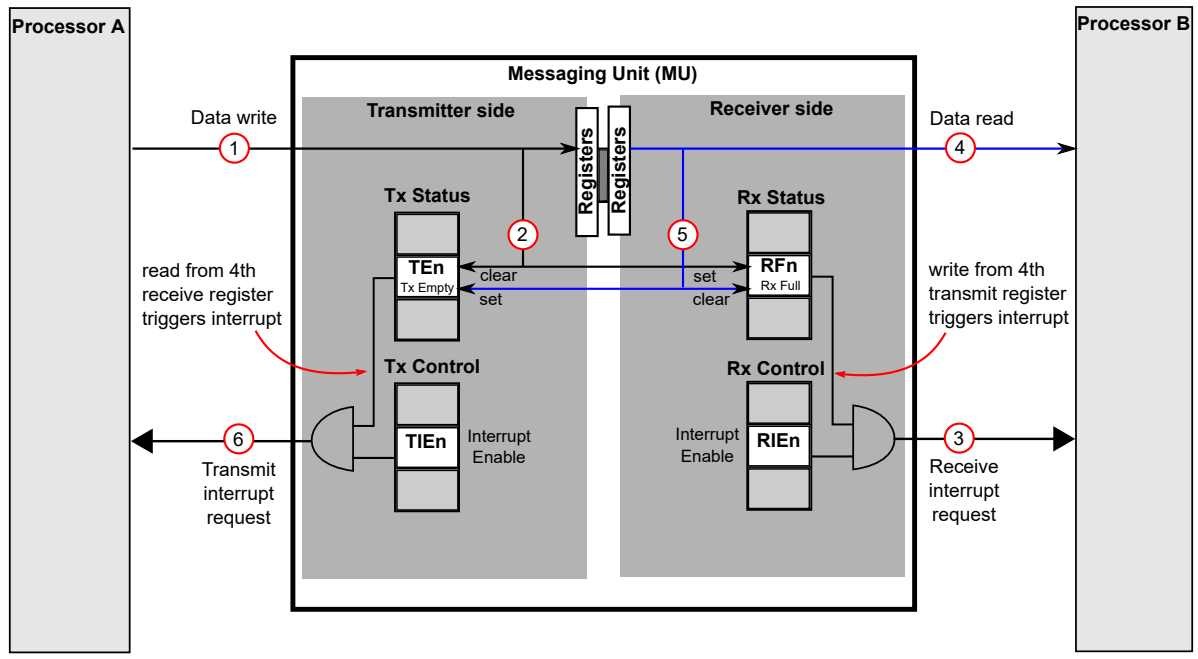


Figure 149. Messaging Model Using Transmit and Receive Registers

The messaging hardware can be used by software to implement messaging protocols for a wide array of message types. Full support is given for both interrupt and polling management schemes.

39.5.2 Messaging Protocols using Event Interrupts

Events and requests that do not include data words can be signaled from the Processor B to the Processor A using the two general interrupts.

Formatted data with a fixed length can be written in predetermined locations in the shared memory. A processor can use a general purpose interrupt to signal the other processor that the data is ready.

The 3 flags can be used by a processor to announce to the other processor the program state it is currently in, or to announce similar messages.

The following table and diagram describe the event sequence when the Processor triggers an interrupt.

Table 201. General Interrupt Messaging Protocol (Generalized)

Sequence	Action	Description
1	Processor A sets General Interrupt request bit	Processor A sets its associated General Interrupt request bit (GIRn = 1) in the General Control Register (GCR).
2	General Interrupt Request Pending status bit is set	The General Interrupt Request Pending status bit (GIPn) in the General Status Register (GSR) is set to "1".
3	General Interrupt request to Processor B is generated	Setting the GIPn bit generates the General Interrupt request to Processor B (Interrupt Request Enable bit, GIEn, must be set for Processor B).
4	Processor B reads status register	The Processor B reads the GIPn bit in the GSR register.
5	Processor B services the interrupt	-
6	Processor B sets GIPn bit to clear interrupt	The Processor B writes "1" to the corresponding GIPn bit to clear the interrupt.
7	GIRn bit is cleared	Setting the GIPn bit to "1" clears the General Interrupt request bit (GIRn) in the Processor A General Control Register (GCR).

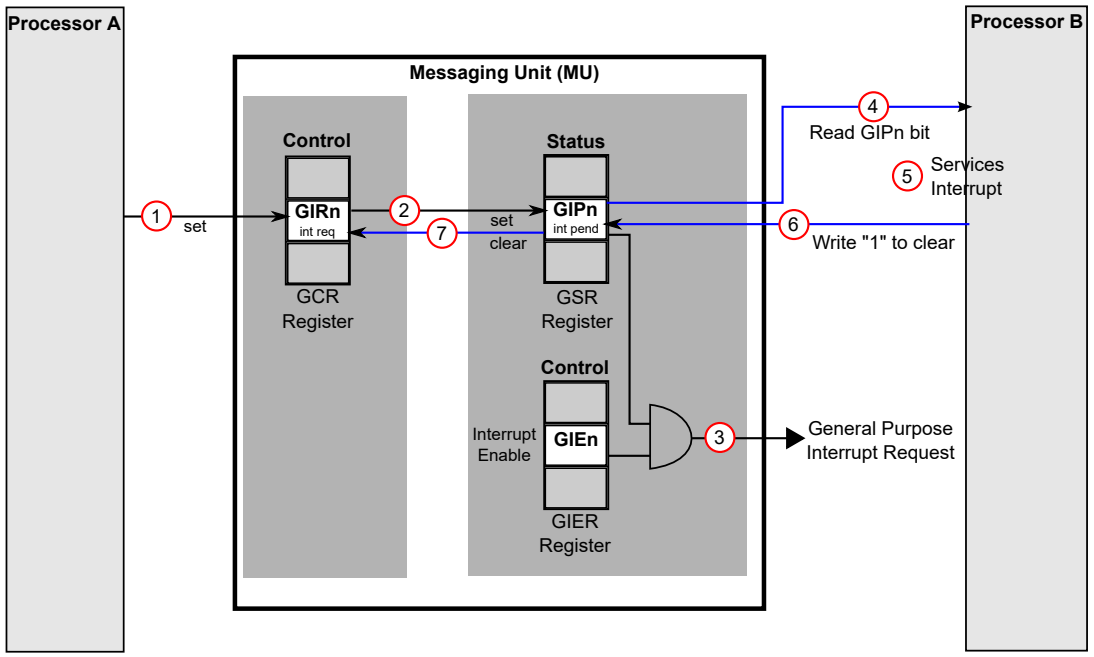


Figure 150. Messaging Model Using a General Purpose Interrupt

39.5.3 Exclusive Access to Shared Memory

MU can be used to signal one processor about its current access to the shared memory, so that the data is not overwritten by the other processor during the exclusive memory access period.

The following tables describe the signaling protocol that the Processor A uses to inform the Processor B about its current access (write) to the shared memory, assuming that the set of bits and registers (GIRn bit, RRn register, TRn register) are reserved to support exclusive access to the shared memory protocol.

Table 202. How the Processor A Performs an Exclusive Access to Shared Memory

Sequence	Action	Description
1	Processor A sends GIRn request to Processor B using Processor A control register	When the Processor A wants to perform an exclusive access to the shared memory, the Processor A sends an GIR0 request to the Processor B.
2	Processor A sends an exclusive-access request using a transmit data register (TRn)	The Processor A sends an exclusive-access request (command, location, and length of target access) to Processor B using a selected transmit data register (TR0).
3	Processor A waits for a dedicated interrupt from Processor B	The Processor A waits for a dedicated interrupt (as an acknowledgement) triggered by the Processor B before proceeding.
4	Processor A accesses shared memory	After receiving a dedicated interrupt from the Processor B, Processor A proceeds.

Table 203. How the Processor B Scans for Transaction Information

Sequence	Action	Description
1	Processor B receives an interrupt from a receive data register (RRn)	-
2	Processor B reads the receive data register (RRn)	-
3	Processor B scans the receive data register contents	For transaction information (whether Processor A has requested an exclusive-access).

Table 204. How the Processor B Accepts Exclusive Access by Processor A

Sequence	Action	Description
1	Processor B triggers a dedicated interrupt	Processor B acknowledges the Processor A request by triggering a dedicated interrupt (ack) to the Processor A.
2	Processor B sends a code message to Processor A	Along with the acknowledge interrupt, the Processor B sends a code message to the Processor A through the selected transmit register (TRn). The message informs the Processor A that it can exclusively access the shared memory.

Table 205. How the Processor B Rejects Exclusive Access by Processor A

Sequence	Action	Description
1	Processor B ignores Processor A request for exclusive access	If the Processor B does not want to give go-ahead permission to the Processor A, Processor B ignores the exclusive access request.

39.5.4 Packet Data Transfers

The following example describes the packet transfer sequence between the Processor B and Processor A subsystems:

Table 206. Packet Data Transfer Sequence

Action	Sequence	Description
Processor B requests DMA	1	The Processor B sends a DMA request to initiate the packet data transfer.
DMA data transfer	2	DMA acknowledges.
	3	DMA starts transferring data from the specified Processor B location to the specified shared memory
	4	DMA interrupts the Processor B to signal that the packet transfer has finished.

Table continues on the next page...

Table 206. Packet Data Transfer Sequence (continued)

Action	Sequence	Description
Processor B informs Processor A that data is in shared memory	5	Using an MU Processor B-side transmit register, the Processor B sends a packet information message to the Processor A to inform the Processor A of the arrival of new packet data that is stored in shared memory. The message contains the command, location, and length of packet data information.
Processor A receives interrupt	6	The Processor A receives an interrupt (assuming its corresponding Processor A MU-side receive interrupt is enabled), and the pending processing task becomes active and processes packet data from memory.
Processor A reads data, writes data	7	The Processor A reads or processes packet data from shared memory.
	8	The Processor A writes the result from packet processing to a separate buffer.
Processor A informs Processor B that transfer is finished	9	After the processing of the packet data finishes, the Processor A informs the Processor B (using the MU Processor A-side transmit register, ATRn).
Processor A sends interrupt to Processor B (request for more data)	10	The Processor B receives the next interrupt from the Processor A, in which the Processor A requests more packet data.

39.5.5 Free the Processor from Deadlock

During normal operations, one processor may conclude that the other processor is not working or is deadlocked. Using the SR register, the following methods are available for the processor to use to identify and correct the problem.

Table 207. How to Free the Processor A/B from Deadlock

Processor Mode	Technique	Description
-	Processor issues an interrupt	The other Processor can interrupt the Processor by issuing any one of the 9 (general purpose, receive, and transmit) interrupts
-	Processor issues a hardware reset	If the above is not helpful, the Processor can issue a hardware reset strobe to the other Processor.

39.6 Register Definition

The MU provides transmit and receive data registers (xTR0-n, xRR0-n) for the communication between Processor A and Processor B. Some control and status registers (xCR, xSR) to the Processor A and Processor B sides for control operations (such as interrupts and reset), and for status checking of the other MU-side. Figure 151 shows the MU registers schematic.

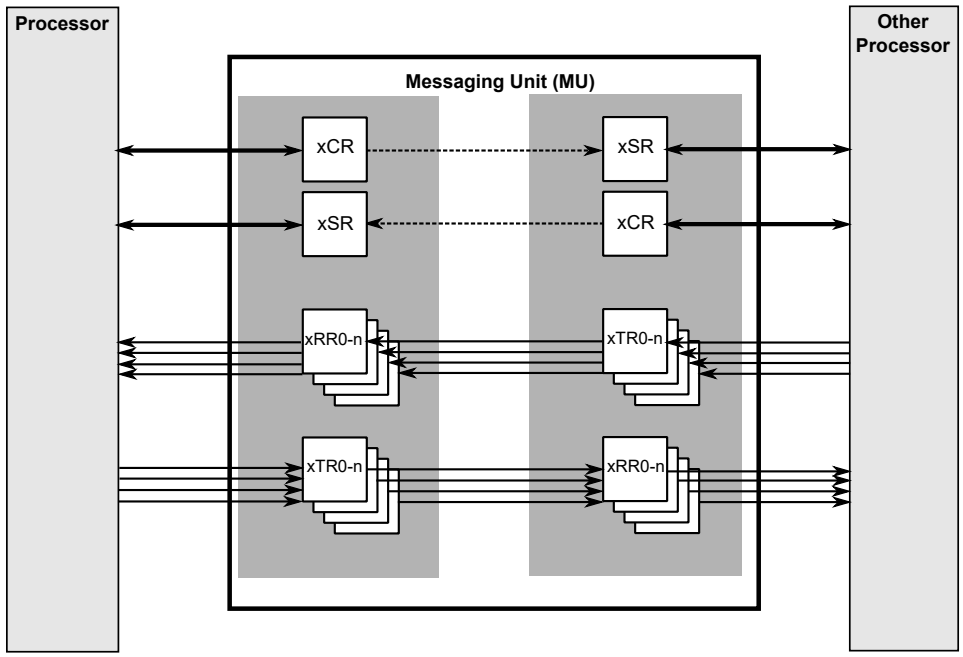


Figure 151. MU Registers

The detailed MU Register Definition can be found below.

39.6.1 MUA register descriptions

This section contains the detailed register descriptions for the MUA registers.

NOTE

A read/write access to any illegal location and a write to a read-only register on the Processor A-side or Processor B-side of the MU generates a module transfer error acknowledge to the Processor A or Processor B. The only exception to this is at address offsets 0x1C and 0x14 - no transfer error occurs when accessing these locations.

39.6.1.1 MUA memory map

MU_2.MUA base address: 400B_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VER)	32	R	0300_000Fh
4h	Parameter Register (PAR)	32	R	0301_0404h
8h	Control Register (CR)	32	RW	0000_0000h
Ch	Status Register (SR)	32	RW	See section
100h	Flag Control Register (FCR)	32	RW	0000_0000h
104h	Flag Status Register (FSR)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
110h	General Interrupt Enable Register (GIER)	32	RW	0000_0000h
114h	General Control Register (GCR)	32	RW	0000_0000h
118h	General Status Register (GSR)	32	RW	0000_0000h
120h	Transmit Control Register (TCR)	32	RW	0000_0000h
124h	Transmit Status Register (TSR)	32	R	0000_000Fh
128h	Receive Control Register (RCR)	32	RW	0000_0000h
12Ch	Receive Status Register (RSR)	32	R	0000_0000h
200h - 20Ch	Transmit Register (TR0 - TR3)	32	RW	0000_0000h
280h - 28Ch	Receive Register (RR0 - RR3)	32	R	0000_0000h

39.6.1.2 Version ID Register (VER)

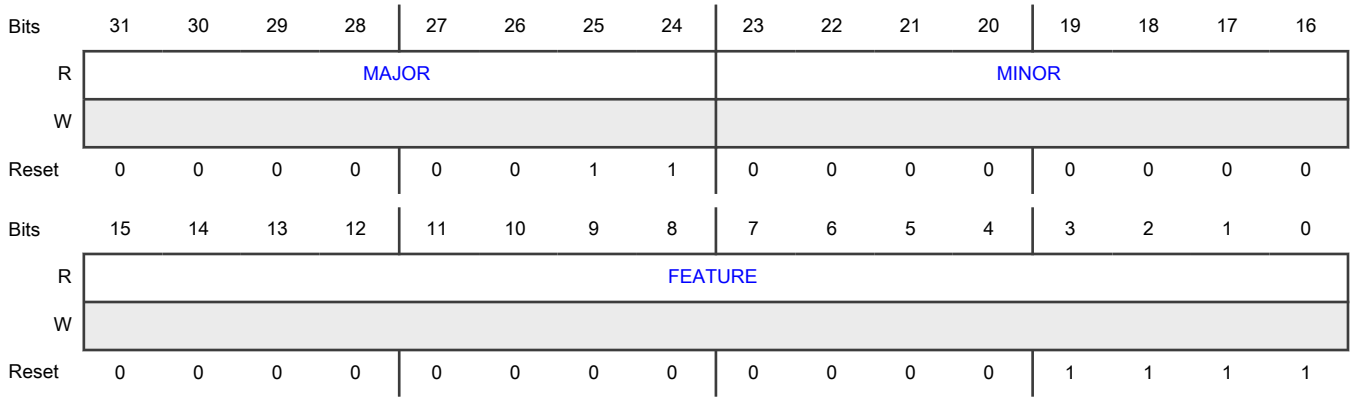
Offset

Register	Offset
VER	0h

Function

Use Version ID register to determine the version ID and feature set number of MUA.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number
23-16 MINOR	Minor Version Number
15-0 FEATURE	Feature Set Number Each bit of the Feature Set Number bitfield specifies a different feature: • If bit0 = 1: Standard features are implemented • If bit1 = 1: RAIP/RAIE register bits are implemented • If bit2 = 1: Core Control and Status Registers are implemented in both MUA and MUB • If bit3 = 1: Expand TRn/RRn registers number

39.6.1.3 Parameter Register (PAR)

Offset

Register	Offset
PAR	4h

Function

The PAR register defines the number of flags, transmit registers, receive registers, and the number of general interrupt requests available for the MU.

Diagram

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

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

Fields

Field	Function
31-24 FLAG_WIDTH	Flag Width This bitfield specifies the number of flag bits in the Flag Control Register (FCR) and Flag Status Register (FSR).
23-16 GIR_NUM	General Interrupt Request Number This bitfield specifies the number of general interrrupt requests available.
15-8 RR_NUM	RR Number This bitfield specifies the number of receive registers.
7-0 TR_NUM	Transmit Register Number This bitfield specifies the number of transmit registers.

39.6.1.4 Control Register (CR)

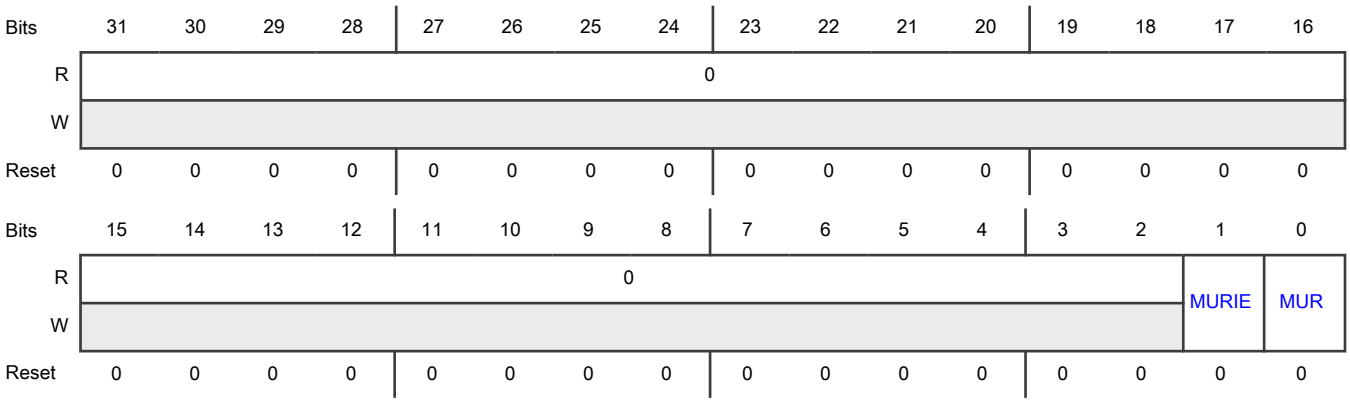
Offset

Register	Offset
CR	8h

Function

The CR register controls the MU reset and MU reset interrupt enable.

Diagram



Fields

Field	Function
31-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 MURIE	MUA Reset Interrupt Enable • If MURIE bit is set to "1", then MU reset interrupt request is issued to the Processor A when the MUA_SR[MURIP] bit is set to "1". • If MURIE is cleared, then the value of the MURIP bit is ignored and no MU reset interrupt request is issued. • Only system reset can reset MURIE, MU reset(MUR) cannot reset this bit. 0b - Disables Processor A-side MU Reset Interrupt request due to MU reset issued by MUB. 1b - Enables Processor A-side MU Reset Interrupt request due to MU reset issued by MUB.
0 MUR	MU Reset • Setting MUR bit to "1" resets both the MUA-side and MUB-side, forcing all control and status registers to return to their default values (except MURIE in MUA/B_CR registers, MURIP bit and MURS bit in MUA/B_SR registers), and all internal states to be cleared. • Before setting the MUR bit to "1", it is advisable to interrupt the Processor B, because setting the MUR bit may affect the ongoing Processor B program. • After setting the MUR bit, monitor the value of the MUA_SR[MURS] bit to know when the reset sequence on the Processor B-side has ended. • MUR bit is always read as "0". • MUR bit is cleared during the MU reset sequence. 0b - Self clearing bit. 1b - Asserts the MU reset.

39.6.1.5 Status Register (SR)

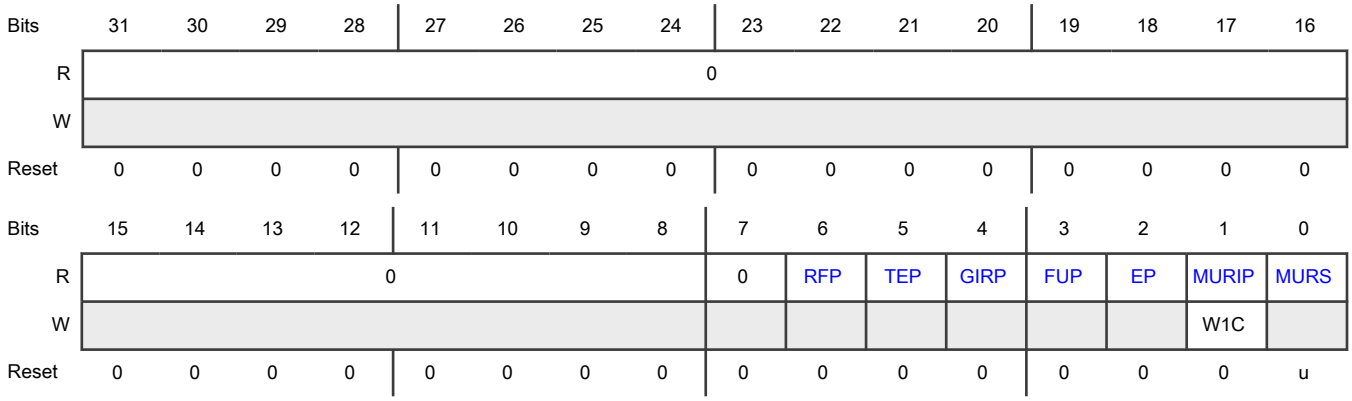
Offset

Register	Offset
SR	Ch

Function

The SR register shows the status of MU resets and the status of pending events/requests described below.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6 RFP	MUA Receive Full Pending Flag • RFP bit is set to "1" when any TRn register is written by MUB to send data to MUA. • After the RFP bit is set to "1", checking the RSR[RFn] bit to decide the data in which RRn register is ready to be read by the MUA. • RFP bit is cleared when all MUA RRn registers are read, or when the MU is reset. 0b - No TRn register is written by MUB. 1b - Any TRn register is written by MUB.
5 TEP	MUA Transmit Empty Pending • TEP bit is set to "1" when TCR[TIEn] bit is set and the corresponding RRn register is read by MUB. • After the TEP bit is set to "1", checking the TSR[TEn] bit to decide the data in which TRn register is ready to be written by the MUA. • TEP bit is cleared when all MUA TRn register are written. • TEP bit is set to "0" when the MU resets. 0b - RRn register is not read by MUB. 1b - Any RRn register is read by MUB.
4 GIRP	MUA General Interrupt Pending • GIRP bit is set to "1" when any general interrupt request is sent from the MUB side to the MUA side. Reading the GSR[GIPn] bit can know which general interrupt request is received.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• GIRP bit is cleared when all GIPn bits in MUA_GSR register are cleared. • GIRP bit is cleared when the MU resets. 0b - No general interrupt request is sent from MUB. 1b - Any general interrupt request is sent from MUB.
3 FUP	MUA Flags Update Pending • FUP bit is set to "1" when the MUA side sends a Flags Update request to the MUB side. • A Flags Update request is generated when there is a change to the Fn[2:0] bits of the MUA_FCR register. No flag update changes are allowed while the FUP bit is set to "1". Any write to the Fn[2:0] bits of the MUA_FCR register, while the FUP bit is set to "1", does not generate a Flags Update event, and the Fn[2:0] bits stays unchanged. • Writing the FCR register doesn't set FUP bit immediately if there is any event pending(SR[EP] is "1"). • FUP bit is cleared when this Flags Update request is internally acknowledged (the flag is updated) from the MUB side, or during MU reset. 0b - No pending update flags(initiated by MUA) are in process 1b - Pending update flags(initiated by MUA) are in process
2 EP	MUA Side Event Pending • An "event" is any hardware message that is reflected in the status register on the MUB side (for example, "transmit register 0 written"). During normal operations, the update mechanism for the EP bit works automatically. • EP bit is set to "1" when the MUA side sends an event update request to the MUB side. • EP bit is cleared by hardware automatically when the event update acknowledge is received. • To ensure events have been posted to MUB, verify that the EP bit is cleared. If EP bit is set to "1", wait and continue to poll the EP bit. • The EP bit is cleared when the MU resets. 0b - The MUA side event is not pending. 1b - The MUA side event is pending.
1 MURIP	MU Reset Interrupt Pending • MURIP bit signals Processor A that Processor B initiated a MU reset by setting the MUB_CR[MUR] bit. • MURIP bit is set to "1" after Processor B initiated a MU reset. If the interrupt is enabled by the MURIE bit, the Processor A MU reset interrupt request is issued when the Processor B set MUR bit in MUB_CR. • Writing "1" to clear the MURIP bit, which also clears MU reset interrupt request. • Only system reset can reset MURIP. MU reset cannot reset this bit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Processor B did not issue MU reset. 1b - Processor B issued MU reset.
0 MURS	MUA and MUB Reset State • The MURS bit is set to "1" during any system reset or any MU reset from MUA-side or MUB-side. • The MURS bit is cleared when the reset sequence on both MUA and MUB side ends. After issuing any of the reset events mentioned previously, **verify that the MURS bit is cleared before starting any access.** 0b - MUA and MUB are out of reset. 1b - MUA or MUB is in reset state.

39.6.1.6 Flag Control Register (FCR)

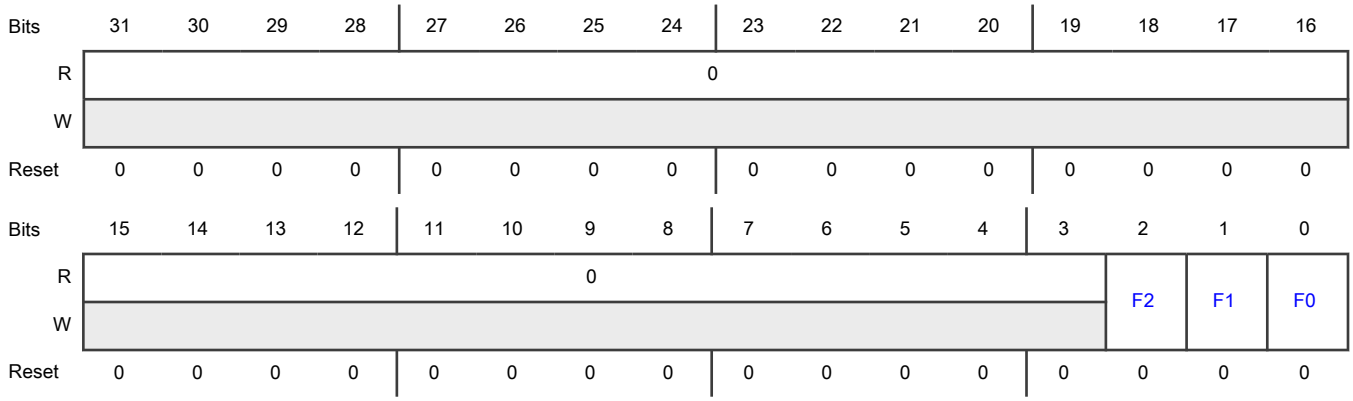
Offset

Register	Offset
FCR	100h

Function

The FCR register contains read-write flags that reflect the MUB_FSR[Fn] bits.

Diagram



Fields

Field	Function
31-3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2-0 Fn	MUA to MUB Flag n • n = 0 to 2 • Fn bit is a read-write flag that reflects the MUB_FSR[Fn] bits. • Fn bits are cleared when the MU resets. • Write 0 to an Fn bit - Clears the Fn bit in the MUB_FSR register. • Write 1 to an Fn bit - Sets the Fn bit in the MUB_FSR register. 0b - Clears the Fn bit in the FSR register. 1b - Sets the Fn bit in the FSR register.

39.6.1.7 Flag Status Register (FSR)

Offset

Register	Offset
FSR	104h

Function

The FSR register contains flags that reflect the values written to the Fn bit of the MUB_FCR register.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													F2	F1	F0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2-0 Fn	MUB to MUA Side Flag n • n = 0 to 2 • Fn bit is the MUA side flag that reflects the values written to the Fn bit in the MUB_FCR register. • Every time that the Fn bit in the MUB_FCR register is written, the Fn bit in the MUB FCR register write event updates the Fn bit in the MUA_FSR register after the event update latency. • If a 0 is read from an Fn bit - Fn bit in the MUB_FCR is 0. • If a 1 is read from an Fn bit - Fn bit in the MUB_FCR is 1. 0b - Fn bit in the MUB FCR register is written 0. 1b - Fn bit in the MUB FCR register is written 1.

39.6.1.8 General Interrupt Enable Register (GIER)

Offset

Register	Offset
GIER	110h

Function

The GIER register contains the MUA general purpose interrupt enables.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															GIE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0-0 GIEn	MUA General Purpose Interrupt Enable n • There are 1 General Purpose Interrupts (n = 0 to 0) • If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor A when the GIPn bit in the MUA GSR register is set to "1". • If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued. • GIEn bit is cleared when the MU resets. 0b - Disables MUA General Interrupt n. 1b - Enables MUA General Interrupt n.

39.6.1.9 General Control Register (GCR)

Offset

Register	Offset
GCR	114h

Function

The GCR register contains the MUA general purpose interrupt requests.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															GIR0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0-0	MUA General Purpose Interrupt Request n

Table continues on the next page...

Table continued from the previous page...

Field	Function
GIRn	• There are 1 General Purpose Interrupts (n = 0 to 0) • Writing "1" to the GIRn bit sets the MUB_GSR[GIPn] bit on the MUB-side. If the MUB_GIER[GIEn] bit is set to "1" on the MUB-side, a General Purpose Interrupt n request is triggered to Processor B. • The GIRn bit is cleared if the MUB_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUA that the interrupt was accepted (cleared by the software). • Writing "0" to the GIRn bit is ignored. • To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit). • GIRn bit is cleared when the MU resets. 0b - MUA General Interrupt n is not requested to the MUB. 1b - MUA General Interrupt n is requested to the MUB.

39.6.1.10 General Status Register (GSR)

Offset

Register	Offset
GSR	118h

Function

The GSR register contains the status of the MUA general interrupt pending requests.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															GIP0
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0-0 GIPn	MUA General Interrupt Request Pending n • There are 1 General Purpose Interrupts (n = 0 to 0) • GIPn bit signals the MUA that the MUB_GCR[GIRn] bit was set from "0" to "1". If the MUA_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor A. • The GIPn bit is cleared by writing "1". Writing "0" is ignored. • GIPn bit is cleared when the MU resets. • After GIPn bit is cleared, the General Interrupt n request(if the MUA_GIER[GIEn] bit is set to "1") is cleared on the MUA side. 0b - MUA general purpose interrupt n is not pending. 1b - MUA general purpose interrupt n is pending.

39.6.1.11 Transmit Control Register (TCR)

Offset

Register	Offset
TCR	120h

Function

The TCR register contains the MUA transmit interrupt enables.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												TIE3	TIE2	TIE1	TIE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 TIEn	MUA Transmit Interrupt Enable n • There are 4 Transmit Interrupts (n = 0 to 3) • If TIEn bit is set to "1", then an MUA Transmit Interrupt n request is issued when the MUA_TSR[TEn] bit is set to "1". • If TIEn bit is cleared, then the value of the TEn bit is ignored and no MUA Transmit Interrupt n request is issued. • TIEn bit is cleared when the MU resets. 0b - Disables MUA Transmit Interrupt n. (default) 1b - Enables MUA Transmit Interrupt n.

39.6.1.12 Transmit Status Register (TSR)

Offset

Register	Offset
TSR	124h

Function

The TSR register shows whether the MUA transmit registers are empty or not.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												TE3	TE2	TE1	TE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Fields

Field	Function
31-4 —	Reserved
3-0 TEn	MUA Transmit Register n Empty • There are 4 Transmit Registers (n = 0 to 3) • The TEn bit is set to "1" after the MUB RRn register is read on the MUB side. • After the TEn bit is set to "1", it signals the MUA side that the MUA TRn register is ready to be written on the MUA side, and a Transmit n interrupt is issued on the MUA side (if the MUA_TCR[TIEn] bit is set to "1"). • TEn bit is cleared after the MUA TRn register is written on the MUA side. • After TEn bit is cleared, the Transmit n interrupt(if the MUA_TCR[TIEn] bit is set to "1") is cleared on the MUA side. • TEn bit is set to "1" when the MU resets. 0b - MUA TRn register is not empty. 1b - MUA TRn register is empty.

39.6.1.13 Receive Control Register (RCR)

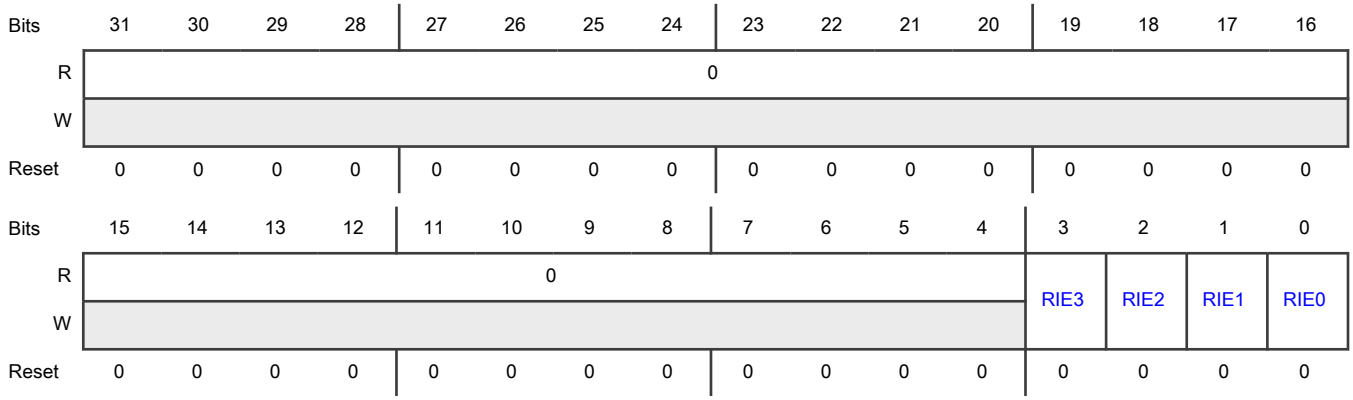
Offset

Register	Offset
RCR	128h

Function

The RCR register contains the MUA receive interrupt enables.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 RIEn	MUA Receive Interrupt Enable n • There are 4 Transmit Interrupts (n = 0 to 3) • If RIEn bit is set to "1", then an MUA Receive Interrupt n request is issued when the MUA_RSR[RFn] bit is set to "1". • If RIEn bit is cleared, then the value of the RFn bit is ignored and no MUA Receive Interrupt n request is issued. • RIEn bit is cleared when the MU resets. 0b - Disables MUA Receive Interrupt n. 1b - Enables MUA Receive Interrupt n.

39.6.1.14 Receive Status Register (RSR)

Offset

Register	Offset
RSR	12Ch

Function

The RSR register shows whether the MUA receive registers are full or not.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												RF3	RF2	RF1	RF0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 RFn	MUA Receive Register n Full • There are 4 Receive Registers (n = 0 to 3) • The RFn bit is set to "1" when the MUB TRn register is written on the MUB side. • After the RFn bit is set to "1", the RFn bit signals the MUA side that new data in the MUA RRn register is ready to be read by the MUA, and a Receive n interrupt is issued on the MUA side (if the MUA_RCR[RIEn] has been set to "1"). • RFn bit is cleared when the MUA RRn register is read, or when the MU is reset. • After RFn bit is cleared, the Receive n interrupt(if the MUA_RCR[RIEn] has been set to "1") is cleared on the MUA side. 0b - MUA RRn register is not full. 1b - MUA RRn register has received data from MUB TRn register and is ready to be read by the MUA.

39.6.1.15 Transmit Register (TR0 - TR3)

Offset

Register	Offset
TR0	200h
TR1	204h
TR2	208h
TR3	20Ch

Function
MUA Transmit Data.

Diagram

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

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

Fields

Field	Function
31-0 TR_DATA	MUA Transmit Data • Data written to the TRn register is reflected in the MUB Receive Register n (RRn). The TRn and RRn registers are not double-buffered, a write to the TRn register overrides the data readable at the RRn register. • A write to the transmit register clears the MUA_TSR[TEn] bit on the transmitter side, and sets the MUB_RSR[RFn] bit on the receiver side. • TRn register can be written only when the MUA_TSR[TEn] bit is set to "1". • Reading the TRn register returns all zeros.

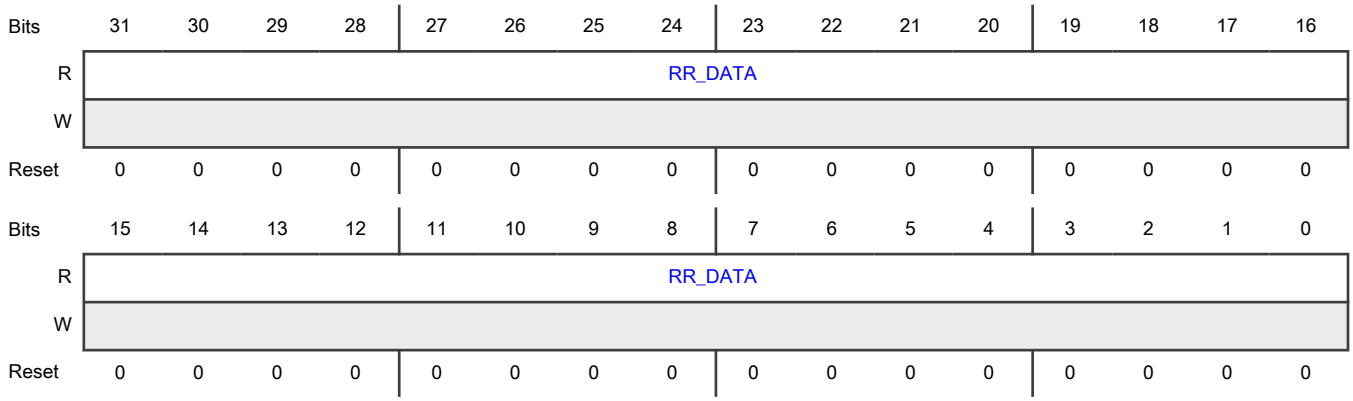
39.6.1.16 Receive Register (RR0 - RR3)

Offset

Register	Offset
RR0	280h
RR1	284h
RR2	288h
RR3	28Ch

Function
MUA Receive Data.

Diagram



Fields

Field	Function
31-0	MUA Receive Data

Table continues on the next page...

Field	Function
RR_DATA	• Reflects the data written to MUB Transmit Register (TRn). • Reading the RRn register clears the MUA_RSR[RFn] bit on the receiver side, and sets the MUB_TSR[TEn] bit on the transmitter side. • RRn register can be read only when the MUB_RSR[RFn] bit is set to "1". Reading before MUB_RSR[RFn]=1 may result in reading incorrect data. Therefore, polling the MUB_RSR[RFn] bit to confirm it is set to "1" before reading RRn is required. • Writing to the RRn register generates an error response to the MUA.

39.6.2 MUB register descriptions

This section contains the detailed register descriptions for the MUB registers.

NOTE

A read/write access to any illegal location and a write to a read-only register on the Processor A-side or Processor B-side of the MU generates a module transfer error acknowledge to the Processor A or Processor B. The only exception to this is at address offsets 0x1C and 0x14 - no transfer error occurs when accessing these locations.

39.6.2.1 MUB memory map

MU_0.MUB base address: 4038_C000h

MU_1.MUB base address: 404E_C000h

MU_2.MUB base address: 400B_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VER)	32	R	0300_000Fh
4h	Parameter Register (PAR)	32	R	See section
8h	Control Register (CR)	32	RW	0000_0000h
Ch	Status Register (SR)	32	RW	See section
10h	Core Control Register 0 (CCR0)	32	RW	0000_0000h
18h	Core Sticky Status Register 0 (CSSR0)	32	RW	0000_0000h
100h	Flag Control Register (FCR)	32	RW	0000_0000h
104h	Flag Status Register (FSR)	32	R	0000_0000h
110h	General Interrupt Enable Register (GIER)	32	RW	0000_0000h
114h	General Control Register (GCR)	32	RW	0000_0000h
118h	General Status Register (GSR)	32	RW	0000_0000h
120h	Transmit Control Register (TCR)	32	RW	0000_0000h
124h	Transmit Status Register (TSR)	32	R	0000_000Fh
128h	Receive Control Register (RCR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
12Ch	Receive Status Register (RSR)	32	R	0000_0000h
200h - 20Ch	Transmit Register (TR0 - TR3)	32	RW	0000_0000h
280h - 28Ch	Receive Register (RR0 - RR3)	32	R	0000_0000h

39.6.2.2 Version ID Register (VER)

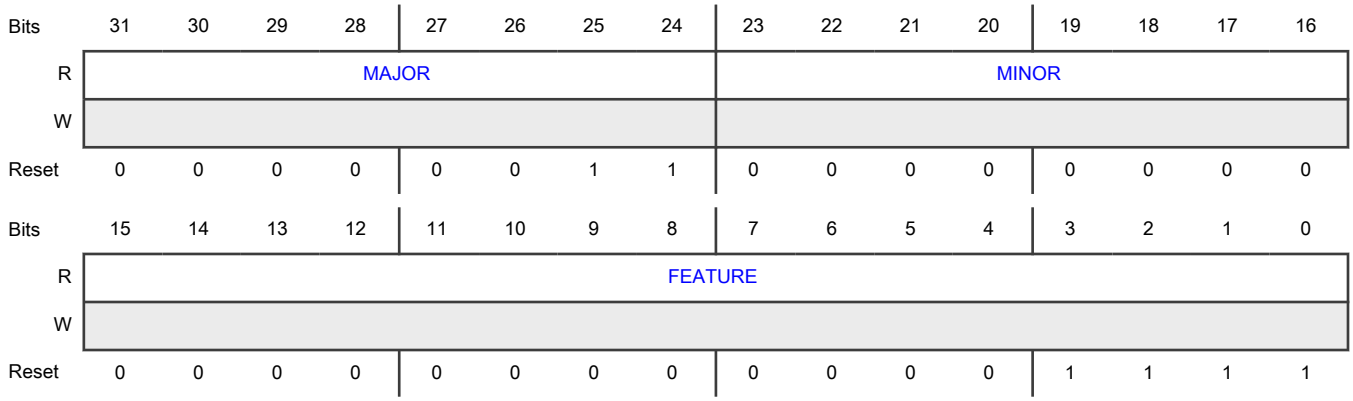
Offset

Register	Offset
VER	0h

Function

Use Version ID register to determine the version ID and feature set number of MUB.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number
23-16 MINOR	Minor Version Number
15-0 FEATURE	Feature Set Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Each bit of the Feature Set Number bitfield specifies a different feature: • If bit0 = 1: Standard features are implemented • If bit1 = 1: RAIP/RAIE register bits are implemented • If bit2 = 1: Core Control and Status Registers are implemented in both MUA and MUB • If bit3 = 1: Expand TRn/RRn registers number

39.6.2.3 Parameter Register (PAR)

Offset

Register	Offset
PAR	4h

Function

The PAR register defines the number of flags, transmit registers, receive registers, and the number of general interrupt requests available for the MU.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FLAG_WIDTH								GIR_NUM							
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RR_NUM								TR_NUM							
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PAR	MU_0.MUB,MU_1.MUB: 2020_0404h MU_2.MUB: 0301_0404h

Fields

Field	Function
31-24 FLAG_WIDTH	Flag Width This bitfield specifies the number of flag bits in the Flag Control Register (FCR) and Flag Status Register (FSR).
23-16 GIR_NUM	General Interrupt Request Number This bitfield specifies the number of general interrrupt requests available.
15-8 RR_NUM	RR Number This bitfield specifies the number of receive registers.
7-0 TR_NUM	Transmit Register Number This bitfield specifies the number of transmit registers.

39.6.2.4 Control Register (CR)

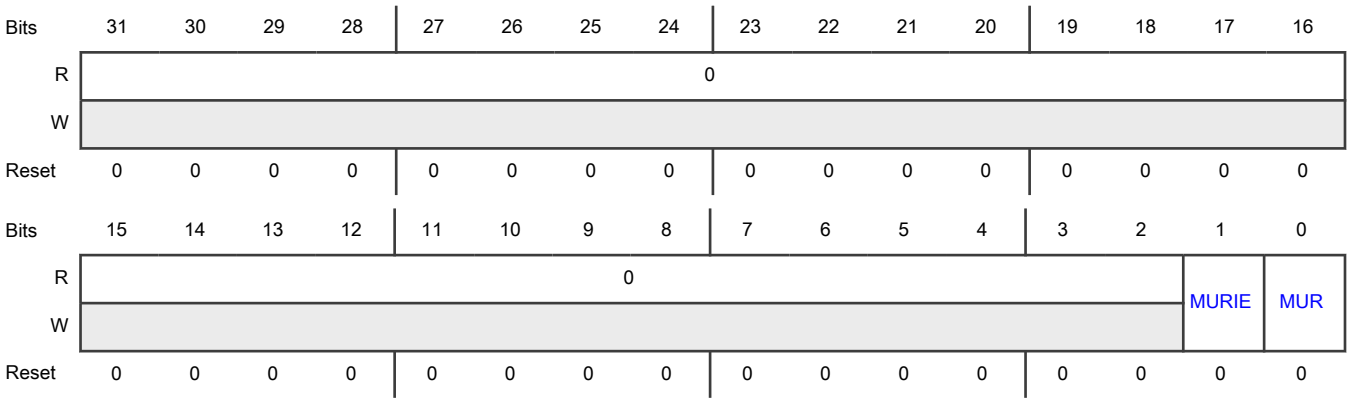
Offset

Register	Offset
CR	8h

Function

The CR register controls the MU reset and MU reset interrupt enable.

Diagram



Fields

Field	Function
31-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 MURIE	MUB Reset Interrupt Enable • If MURIE bit is set to "1", then MU reset interrupt request is issued to the Processor B when the MUB_SR[MURIP] bit is set to "1". • If MURIE is cleared, then the value of the MURIP bit is ignored and no MU reset interrupt request is issued. • Only system reset can reset MURIE, MU reset(MUR) cannot reset this bit. 0b - Disables Processor B-side MU Reset Interrupt request due to MU reset issued by MUA. 1b - Enables Processor B-side MU Reset Interrupt request due to MU reset issued by MUA.
0 MUR	MU Reset • Setting MUR bit to "1" resets both the MUB-side and MUA-side, forcing all control and status registers to return to their default values (except in MUB/A_CCR0 registers, MURIE in MUB/A_CR registers, MURIP bit and MURS bit in MUB/A_SR registers), and all internal states to be cleared. • Before setting the MUR bit to "1", it is advisable to interrupt the Processor A, because setting the MUR bit may affect the ongoing Processor A program. • After setting the MUR bit, monitor the value of the MUB_SR[MURS] bit to know when the reset sequence on the Processor A-side has ended. • MUR bit is always read as "0". • MUR bit is cleared during the MU reset sequence. 0b - Self clearing bit. 1b - Asserts the MU reset.

39.6.2.5 Status Register (SR)

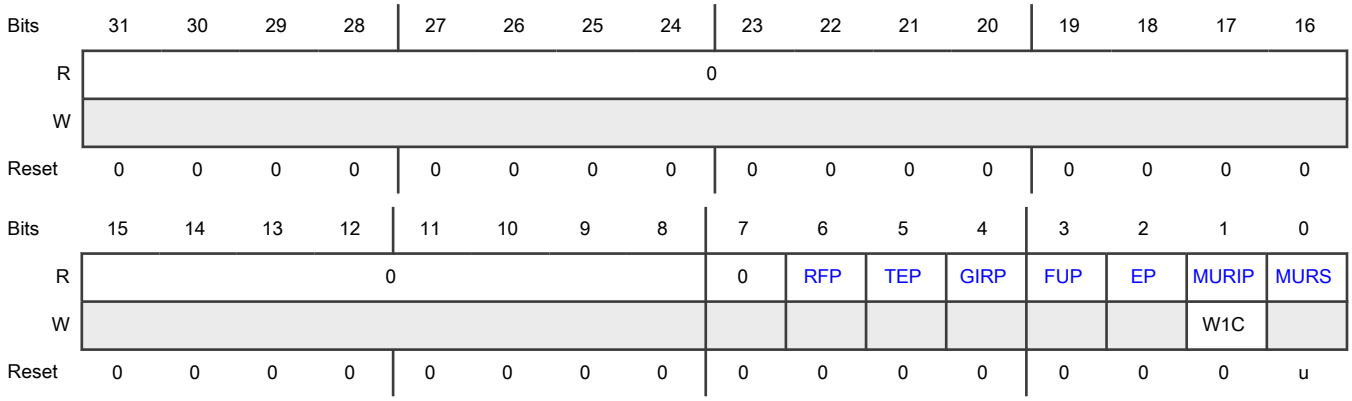
Offset

Register	Offset
SR	Ch

Function

The SR register shows the status of MU resets and the status of pending events/requests described below.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6 RFP	MUB Receive Full Pending Flag • RFP bit is set to "1" when any TRn register is written by MUA to send data to MUB. • After the RFP bit is set to "1", checking the RSR[RFn] bit to decide the data in which RRn register is ready to be read by the MUB. • RFP bit is cleared when all MUB RRn registers are read, or when the MU is reset. 0b - No TRn register is written by MUA. 1b - Any TRn register is written by MUA.
5 TEP	MUB Transmit Empty Pending • TEP bit is set to "1" when TCR[TIEn] bit is set and the corresponding RRn register is read by MUA. • After the TEP bit is set to "1", checking the TSR[TEn] bit to decide the data in which TRn register is ready to be written by the MUB. • TEP bit is cleared when all MUB TRn register are written. • TEP bit is set to "0" when the MU resets. 0b - RRn register is not read by MUA. 1b - Any RRn register is read by MUA.
4 GIRP	MUB General Interrupt Pending • GIRP bit is set to "1" when any general interrupt request is sent from the MUA side to the MUB side. Reading the GSR[GIPn] bit can know which general interrupt request is received.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• GIRP bit is cleared when all GIPn bits in MUB_GSR register are cleared. • GIRP bit is cleared when the MU resets. 0b - No general interrupt request is sent from MUA. 1b - Any general interrupt request is sent from MUA.
3 FUP	MUB Flags Update Pending • FUP bit is set to "1" when the MUB side sends a Flags Update request to the MUA side. • A Flags Update request is generated when there is a change to the Fn[31:0] bits of the MUB_FCR register. No flag update changes are allowed while the FUP bit is set to "1". Any write to the Fn[31:0] bits of the MUB_FCR register, while the FUP bit is set to "1", does not generate a Flags Update event, and the Fn[31:0] bits stays unchanged. • Writing the FCR register doesn't set FUP bit immediately if there is any event pending(SR[EP] is "1"). • FUP bit is cleared when this Flags Update request is internally acknowledged (the flag is updated) from the MUA side, or during MU reset. 0b - No pending update flags(initiated by MUB) are in process 1b - Pending update flags(initiated by MUB) are in process
2 EP	MUB Side Event Pending • An "event" is any hardware message that is reflected in the status register on the MUA side (for example, "transmit register 0 written"). During normal operations, the update mechanism for the EP bit works automatically. • EP bit is set to "1" when the MUB side sends an event update request to the MUA side. • EP bit is cleared by hardware automatically when the event update acknowledge is received. • To ensure events have been posted to MUA, verify that the EP bit is cleared. If EP bit is set to "1", wait and continue to poll the EP bit. • The EP bit is cleared when the MU resets. 0b - The MUB side event is not pending. 1b - The MUB side event is pending.
1 MURIP	MU Reset Interrupt Pending • MURIP bit signals Processor B that Processor A initiated a MU reset by setting the MUA_CR[MUR] bit. • MURIP bit is set to "1" after Processor A initiated a MU reset. If the interrupt is enabled by the MURIE bit, the Processor B MU reset interrupt request is issued when the Processor A set MUR bit in MUA_CR. • Writing "1" to clear the MURIP bit, which also clears MU reset interrupt request. • Only system reset can reset MURIP. MU reset cannot reset this bit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Processor A did not issue MU reset. 1b - Processor A issued MU reset.
0 MURS	MUA and MUB Reset State • The MURS bit is set to "1" during any system reset or any MU reset from MUA-side or MUB-side. • The MURS bit is cleared when the reset sequence on both MUA and MUB side ends. After issuing any of the reset events mentioned previously, **verify that the MURS bit is cleared before starting any access.** 0b - MUA and MUB are out of reset. 1b - MUA or MUB is in reset state.

39.6.2.6 Core Control Register 0 (CCR0)

Offset

Register	Offset
CCR0	10h

Function

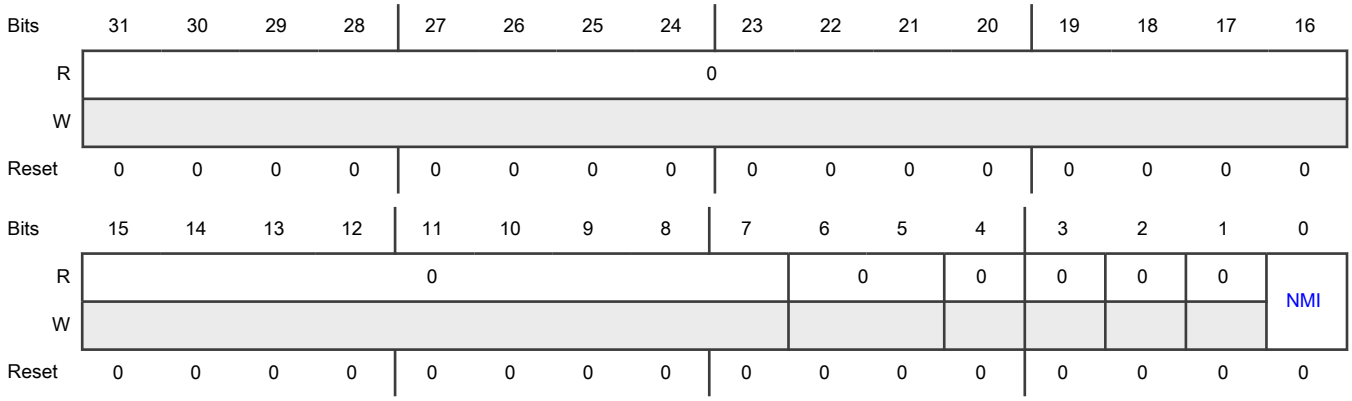
The CCR0 enables the MUB to control the Processor on the MUA-side.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
MU_0.MUB	CCR0	—
MU_1.MUB	CCR0	—
MU_2.MUB	—	CCR0

Diagram



Fields

Field	Function
31-7 —	Reserved
6-5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 NMI	MUA Non-maskable Interrupt Request • When NMI bit is set to "1", it initiates a Non-Maskable Interrupt to the Processor A. • NMI bit is cleared after the MUA_CSSR0[NMIC] bit is written to "1". After the NMI bit is cleared, the MUB can initiate another non-maskable interrupt to the MUA. • The NMI bit is cleared when the MU resets. 0b - Non-maskable interrupt is not issued to the Processor A by the Processor B. 1b - Non-maskable interrupt is issued to the Processor A by the Processor B.

39.6.2.7 Core Sticky Status Register 0 (CSSR0)

Offset

Register	Offset
CSSR0	18h

Function

The CSSR0 register shows the status of interrupts pending (W1C).

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
MU_0.MUB	CSSR0	—
MU_1.MUB	CSSR0	—
MU_2.MUB	—	CSSR0

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	0	0	0	0	0	0	NMIC
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 NMIC	Processor B Non-Maskable-Interrupt Clear • NMIC bit is used by the MUB-side Non-Maskable Interrupt service routine to clear the non-maskable interrupt from MUA-side. • Writing NMIC bit as "1" signals the MUA to clear the MUA_CCR0[NMI] bit, thus de-asserting the interrupt and enabling the NMI bit to receive another interrupt. • The NMIC bit is always read as "0", therefore NMIC bit cannot be polled. The NMIC bit can only be used as part of the interrupt service routine, in which the NMIC bit should only be written as "1" once. • The NMIC bit is cleared when the MU resets. 0b - Default 1b - Writing "1" clears the MUA_CCR0[NMI] bit.

39.6.2.8 Flag Control Register (FCR)

Offset

Register	Offset
FCR	100h

Function

The FCR register contains read-write flags that reflect the MUA_FSR[Fn] bits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	F31	F30	F29	F28	F27	F26	F25	F24	F23	F22	F21	F20	F19	F18	F17	F16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31
F31</td><td>MUB to MUA Flag n
• n = 0 to 31
• Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
• Fn bits are cleared when the MU resets.
• Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
• Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr></table>0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.</td></tr>
<tr><td>30
F30</td><td>MUB to MUA Flag n
• n = 0 to 31
• Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
• Fn bits are cleared when the MU resets.
• Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
• Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>29
F29</td><td>
MUB to MUA Flag n

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>28
F28</td><td>
MUB to MUA Flag n

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>27
F27</td><td>MUB to MUA Flag n

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>26</td><td>MUB to MUA Flag n</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>F26</td><td>

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>25
F25</td><td>MUB to MUA Flag n

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>1b - Sets the Fn bit in the FSR register.</td></tr>
<tr><td>24
F24</td><td>MUB to MUA Flag n
• n = 0 to 31
• Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
• Fn bits are cleared when the MU resets.
• Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
• Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr></table>0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.</td></tr>
<tr><td>23
F23</td><td>MUB to MUA Flag n
• n = 0 to 31
• Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
• Fn bits are cleared when the MU resets.
• Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
• Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>22
F22</td><td>MUB to MUA Flag n
• n = 0 to 31
• Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
• Fn bits are cleared when the MU resets.
• Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
• Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>21
F21</td><td>MUB to MUA Flag n
• n = 0 to 31
• Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
• Fn bits are cleared when the MU resets.
• Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
• Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>20
F20</td><td>MUB to MUA Flag n
• n = 0 to 31
• Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
• Fn bits are cleared when the MU resets.
• Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
• Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>19
F19</td><td>MUB to MUA Flag n
• n = 0 to 31
• Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
• Fn bits are cleared when the MU resets.
• Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
• Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>18
F18</td><td>
MUB to MUA Flag n

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>17
F17</td><td>
MUB to MUA Flag n

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>16
F16</td><td>MUB to MUA Flag n

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>15</td><td>MUB to MUA Flag n</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>F15</td><td>

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>14
F14</td><td>MUB to MUA Flag n

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>1b - Sets the Fn bit in the FSR register.</td></tr>
<tr><td>13
F13</td><td>MUB to MUA Flag n
• n = 0 to 31
• Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
• Fn bits are cleared when the MU resets.
• Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
• Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr></table>0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.</td></tr>
<tr><td>12
F12</td><td>MUB to MUA Flag n
• n = 0 to 31
• Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
• Fn bits are cleared when the MU resets.
• Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
• Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>11
F11</td><td>MUB to MUA Flag n
• n = 0 to 31
• Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
• Fn bits are cleared when the MU resets.
• Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
• Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>10
F10</td><td>MUB to MUA Flag n
• n = 0 to 31
• Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
• Fn bits are cleared when the MU resets.
• Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
• Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr></table> 0b - Clears the Fn bit in the FSR register. 1b - Sets the Fn bit in the FSR register.
9 F9	MUB to MUA Flag n • n = 0 to 31 • Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits. • Fn bits are cleared when the MU resets. • Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register. • Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr></table> 0b - Clears the Fn bit in the FSR register. 1b - Sets the Fn bit in the FSR register.
8 F8	MUB to MUA Flag n • n = 0 to 31 • Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits. • Fn bits are cleared when the MU resets. • Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register. • Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>7
F7</td><td>
MUB to MUA Flag n

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>6
F6</td><td>
MUB to MUA Flag n

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>5
F5</td><td>MUB to MUA Flag n

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>4</td><td>MUB to MUA Flag n</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>F4</td><td>

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
1b - Sets the Fn bit in the FSR register.
</td></tr>
<tr><td>3
F3</td><td>MUB to MUA Flag n

n = 0 to 31
Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.
Fn bits are cleared when the MU resets.
Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register.
Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clears the Fn bit in the FSR register.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Sets the Fn bit in the FSR register.
2 F2	MUB to MUA Flag n • n = 0 to 31 • Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits. • Fn bits are cleared when the MU resets. • Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register. • Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register. 0b - Clears the Fn bit in the FSR register. 1b - Sets the Fn bit in the FSR register.
1 F1	MUB to MUA Flag n • n = 0 to 31 • Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits. • Fn bits are cleared when the MU resets. • Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register. • Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register. 0b - Clears the Fn bit in the FSR register. 1b - Sets the Fn bit in the FSR register.
0 F0	MUB to MUA Flag n • n = 0 to 31 • Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits. • Fn bits are cleared when the MU resets. • Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register. • Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register. 0b - Clears the Fn bit in the FSR register. 1b - Sets the Fn bit in the FSR register.

39.6.2.9 Flag Status Register (FSR)

Offset

Register	Offset
FSR	104h

Function

The FSR register contains flags that reflect the values written to the Fn bit of the MUA_FCR register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	F31	F30	F29	F28	F27	F26	F25	F24	F23	F22	F21	F20	F19	F18	F17	F16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31
F31</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr></table>0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.</td></tr>
<tr><td>30
F30</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>29
F29</td><td>MUA to MUB Side Flag n

n = 0 to 31
Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>28</td><td>MUA to MUB Side Flag n</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>F28</td><td>

n = 0 to 31
Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>27
F27</td><td>MUA to MUB Side Flag n

n = 0 to 31
Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.</td></tr>
<tr><td>26
F26</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr></table>0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.</td></tr>
<tr><td>25
F25</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>24
F24</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>23
F23</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>22
F22</td><td>
MUA to MUB Side Flag n

n = 0 to 31
Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>21
F21</td><td>
MUA to MUB Side Flag n

n = 0 to 31
Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>20
F20</td><td>MUA to MUB Side Flag n

n = 0 to 31
Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>19
F19</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr></table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.</td></tr>
<tr><td>18
F18</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>17
F17</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>16
F16</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>15
F15</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>14
F14</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>13
F13</td><td>
MUA to MUB Side Flag n

n = 0 to 31
Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>12
F12</td><td>
MUA to MUB Side Flag n
n = 0 to 31
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>11
F11</td><td>
MUA to MUB Side Flag n

n = 0 to 31
Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>1b - Fn bit in the MUA FCR register is written 1.</td></tr>
<tr><td>10
F10</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr></table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.</td></tr>
<tr><td>9
F9</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>8
F8</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>7
F7</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>6
F6</td><td>
MUA to MUB Side Flag n

n = 0 to 31
Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>5
F5</td><td>
MUA to MUB Side Flag n

n = 0 to 31
Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
<tr><td>4
F4</td><td>
MUA to MUB Side Flag n

n = 0 to 31
Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>3
F3</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr></table>
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.</td></tr>
<tr><td>2
F2</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.
• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1.
0b - Fn bit in the MUA FCR register is written 0.
1b - Fn bit in the MUA FCR register is written 1.</td></tr>
<tr><td>1
F1</td><td>MUA to MUB Side Flag n
• n = 0 to 31
• Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register.
• Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency.
• If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1. 0b - Fn bit in the MUA FCR register is written 0. 1b - Fn bit in the MUA FCR register is written 1.
0 F0	MUA to MUB Side Flag n • n = 0 to 31 • Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register. • Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency. • If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0. • If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1. 0b - Fn bit in the MUA FCR register is written 0. 1b - Fn bit in the MUA FCR register is written 1.

39.6.2.10 General Interrupt Enable Register (GIER)

Offset

Register	Offset
GIER	110h

Function

The GIER register contains the MUB general purpose interrupt enables.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GIE31	GIE30	GIE29	GIE28	GIE27	GIE26	GIE25	GIE24	GIE23	GIE22	GIE21	GIE20	GIE19	GIE18	GIE17	GIE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GIE15	GIE14	GIE13	GIE12	GIE11	GIE10	GIE9	GIE8	GIE7	GIE6	GIE5	GIE4	GIE3	GIE2	GIE1	GIE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31
GIE31</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr></table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>30
GIE30</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr></table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.</td></tr>
<tr><td>29
GIE29</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr></table>
0b - Disables MUB General Interrupt n.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>1b - Enables MUB General Interrupt n.</td></tr>
<tr><td>28
GIE28</td><td>MUB General Purpose Interrupt Enable n
There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr></table>0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.</td></tr>
<tr><td>27
GIE27</td><td>MUB General Purpose Interrupt Enable n
There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>26
GIE26</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>25
GIE25</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>24
GIE24</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>23
GIE23</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>22
GIE22</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>21
GIE21</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>20
GIE20</td><td>
MUB General Purpose Interrupt Enable n

There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>19
GIE19</td><td>
MUB General Purpose Interrupt Enable n

There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>18
GIE18</td><td>
MUB General Purpose Interrupt Enable n

There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>17
GIE17</td><td>
MUB General Purpose Interrupt Enable n

There are 32 General Purpose Interrupts (n = 0 to 31)

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>16
GIE16</td><td>
MUB General Purpose Interrupt Enable n

There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>15</td><td>MUB General Purpose Interrupt Enable n</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>GIE15</td><td>

There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>14
GIE14</td><td>MUB General Purpose Interrupt Enable n

There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>13
GIE13</td><td>MUB General Purpose Interrupt Enable n
There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr></table>0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.</td></tr>
<tr><td>12
GIE12</td><td>MUB General Purpose Interrupt Enable n
There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr></table>0b - Disables MUB General Interrupt n.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>1b - Enables MUB General Interrupt n.</td></tr>
<tr><td>11
GIE11</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr></table>0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.</td></tr>
<tr><td>10
GIE10</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>9
GIE9</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>8
GIE8</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>7
GIE7</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>6
GIE6</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>5
GIE5</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>4
GIE4</td><td>MUB General Purpose Interrupt Enable n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
• GIEn bit is cleared when the MU resets.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>3
GIE3</td><td>
MUB General Purpose Interrupt Enable n

There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>2
GIE2</td><td>
MUB General Purpose Interrupt Enable n

There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>1
GIE1</td><td>
MUB General Purpose Interrupt Enable n

There are 32 General Purpose Interrupts (n = 0 to 31)
If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1".
If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued.
GIEn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disables MUB General Interrupt n.
1b - Enables MUB General Interrupt n.
</td></tr>
<tr><td>0
GIE0</td><td>
MUB General Purpose Interrupt Enable n

There are 32 General Purpose Interrupts (n = 0 to 31)

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1". • If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request is issued. • GIEn bit is cleared when the MU resets. 0b - Disables MUB General Interrupt n. 1b - Enables MUB General Interrupt n.

39.6.2.11 General Control Register (GCR)

Offset

Register	Offset
GCR	114h

Function

The GCR register contains the MUB general purpose interrupt requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GIR31	GIR30	GIR29	GIR28	GIR27	GIR26	GIR25	GIR24	GIR23	GIR22	GIR21	GIR20	GIR19	GIR18	GIR17	GIR16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GIR15	GIR14	GIR13	GIR12	GIR11	GIR10	GIR9	GIR8	GIR7	GIR6	GIR5	GIR4	GIR3	GIR2	GIR1	GIR0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 GIR31	MUB General Purpose Interrupt Request n • There are 32 General Purpose Interrupts (n = 0 to 31) • Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A. • The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>30
GIR30</td><td>
MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>29
GIR29</td><td>MUB General Purpose Interrupt Request n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
• The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
• Writing "0" to the GIRn bit is ignored.
• To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
• GIRn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.</td></tr>
<tr><td>28
GIR28</td><td>MUB General Purpose Interrupt Request n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
• The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
• Writing "0" to the GIRn bit is ignored.
• To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
GIRn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>27
GIR27</td><td>
MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>26
GIR26</td><td>MUB General Purpose Interrupt Request n
There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr></table>0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.</td></tr>
<tr><td>25
GIR25</td><td>MUB General Purpose Interrupt Request n
There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>24
GIR24</td><td>MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>23
GIR23</td><td>MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>22
GIR22</td><td>
MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>21
GIR21</td><td>MUB General Purpose Interrupt Request n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
• The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
• Writing "0" to the GIRn bit is ignored.
• To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
• GIRn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.</td></tr>
<tr><td>20
GIR20</td><td>MUB General Purpose Interrupt Request n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
• The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
• Writing "0" to the GIRn bit is ignored.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>19
GIR19</td><td>
MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.</td></tr>
<tr><td>18
GIR18</td><td>MUB General Purpose Interrupt Request n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
• The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
• Writing "0" to the GIRn bit is ignored.
• To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
• GIRn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr></table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.</td></tr>
<tr><td>17
GIR17</td><td>MUB General Purpose Interrupt Request n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
• The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
• Writing "0" to the GIRn bit is ignored.
• To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
• GIRn bit is cleared when the MU resets.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>16
GIR16</td><td>
MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>15</td><td>MUB General Purpose Interrupt Request n</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>GIR15</td><td>

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>14
GIR14</td><td>MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>13
GIR13</td><td>MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>12
GIR12</td><td>MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>11
GIR11</td><td>
MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>10
GIR10</td><td>MUB General Purpose Interrupt Request n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
• The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
• Writing "0" to the GIRn bit is ignored.
• To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
• GIRn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>9
GIR9</td><td>MUB General Purpose Interrupt Request n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
• The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
• Writing "0" to the GIRn bit is ignored.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>8
GIR8</td><td>
MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.</td></tr>
<tr><td>7
GIR7</td><td>MUB General Purpose Interrupt Request n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
• The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
• Writing "0" to the GIRn bit is ignored.
• To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
• GIRn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr></table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.</td></tr>
<tr><td>6
GIR6</td><td>MUB General Purpose Interrupt Request n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
• The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
• Writing "0" to the GIRn bit is ignored.
• To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
• GIRn bit is cleared when the MU resets.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>5
GIR5</td><td>
MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>4</td><td>MUB General Purpose Interrupt Request n</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>GIR4</td><td>

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>3
GIR3</td><td>MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>2
GIR2</td><td>MUB General Purpose Interrupt Request n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
• The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
• Writing "0" to the GIRn bit is ignored.
• To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
• GIRn bit is cleared when the MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>1
GIR1</td><td>MUB General Purpose Interrupt Request n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.
</td></tr>
<tr><td>0
GIR0</td><td>MUB General Purpose Interrupt Request n

There are 32 General Purpose Interrupts (n = 0 to 31)
Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A.
The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software).
Writing "0" to the GIRn bit is ignored.
To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit).
GIRn bit is cleared when the MU resets.

0b - MUB General Interrupt n is not requested to the MUA.
1b - MUB General Interrupt n is requested to the MUA.</td></tr>
</table>

39.6.2.12 General Status Register (GSR)

Offset

Register	Offset
GSR	118h

Function

The GSR register contains the status of the MUB general interrupt pending requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GIP31	GIP30	GIP29	GIP28	GIP27	GIP26	GIP25	GIP24	GIP23	GIP22	GIP21	GIP20	GIP19	GIP18	GIP17	GIP16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GIP15	GIP14	GIP13	GIP12	GIP11	GIP10	GIP9	GIP8	GIP7	GIP6	GIP5	GIP4	GIP3	GIP2	GIP1	GIP0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31
GIP31</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr></table>0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.</td></tr>
<tr><td>30
GIP30</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>29
GIP29</td><td>
MUB General Interrupt Request Pending n

There are 32 General Purpose Interrupts (n = 0 to 31)
GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>28
GIP28</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.</td></tr>
<tr><td>27
GIP27</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>26
GIP26</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>25
GIP25</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>24
GIP24</td><td>
MUB General Interrupt Request Pending n

There are 32 General Purpose Interrupts (n = 0 to 31)
GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>23
GIP23</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr></table>0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.</td></tr>
<tr><td>22
GIP22</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>21
GIP21</td><td>
MUB General Interrupt Request Pending n

There are 32 General Purpose Interrupts (n = 0 to 31)
GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>20
GIP20</td><td>
MUB General Interrupt Request Pending n

There are 32 General Purpose Interrupts (n = 0 to 31)
GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>19
GIP19</td><td>
MUB General Interrupt Request Pending n

There are 32 General Purpose Interrupts (n = 0 to 31)
GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>18
GIP18</td><td>
MUB General Interrupt Request Pending n

There are 32 General Purpose Interrupts (n = 0 to 31)

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>17
GIP17</td><td>
MUB General Interrupt Request Pending n

There are 32 General Purpose Interrupts (n = 0 to 31)
GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>16
GIP16</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.</td></tr>
<tr><td>15
GIP15</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>14
GIP14</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>13
GIP13</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>12
GIP12</td><td>
MUB General Interrupt Request Pending n

There are 32 General Purpose Interrupts (n = 0 to 31)
GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>11
GIP11</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr></table>0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.</td></tr>
<tr><td>10
GIP10</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>9
GIP9</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>8
GIP8</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>7
GIP7</td><td>
MUB General Interrupt Request Pending n

There are 32 General Purpose Interrupts (n = 0 to 31)
GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>6
GIP6</td><td>
MUB General Interrupt Request Pending n

There are 32 General Purpose Interrupts (n = 0 to 31)

</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>5
GIP5</td><td>
MUB General Interrupt Request Pending n

There are 32 General Purpose Interrupts (n = 0 to 31)
GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>4
GIP4</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.</td></tr>
<tr><td>3
GIP3</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>2
GIP2</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.
• GIPn bit is cleared when the MU resets.
• After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>1
GIP1</td><td>MUB General Interrupt Request Pending n
• There are 32 General Purpose Interrupts (n = 0 to 31)
• GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
• The GIPn bit is cleared by writing "1". Writing "0" is ignored.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table>
0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.
</td></tr>
<tr><td>0
GIP0</td><td>MUB General Interrupt Request Pending n

There are 32 General Purpose Interrupts (n = 0 to 31)
GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B.
The GIPn bit is cleared by writing "1". Writing "0" is ignored.
GIPn bit is cleared when the MU resets.
After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side.

0b - MUB general purpose interrupt n is not pending.
1b - MUB general purpose interrupt n is pending.</td></tr>
</table>

39.6.2.13 Transmit Control Register (TCR)

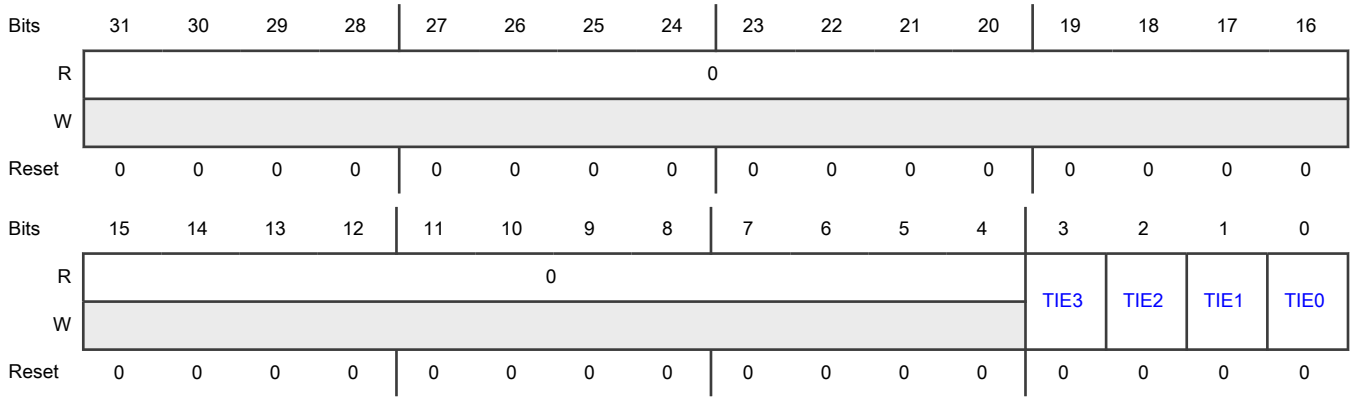
Offset

Register	Offset
TCR	120h

Function

The TCR register contains the MUB transmit interrupt enables.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TIEn	MUB Transmit Interrupt Enable n • There are 4 Transmit Interrupts (n = 0 to 3) • If TIEn bit is set to "1", then an MUB Transmit Interrupt n request is issued when the MUB_TSR[TEn] bit is set to "1". • If TIEn bit is cleared, then the value of the TEn bit is ignored and no MUB Transmit Interrupt n request is issued. • TIEn bit is cleared when the MU resets. 0b - Disables MUB Transmit Interrupt n. (default) 1b - Enables MUB Transmit Interrupt n.

39.6.2.14 Transmit Status Register (TSR)

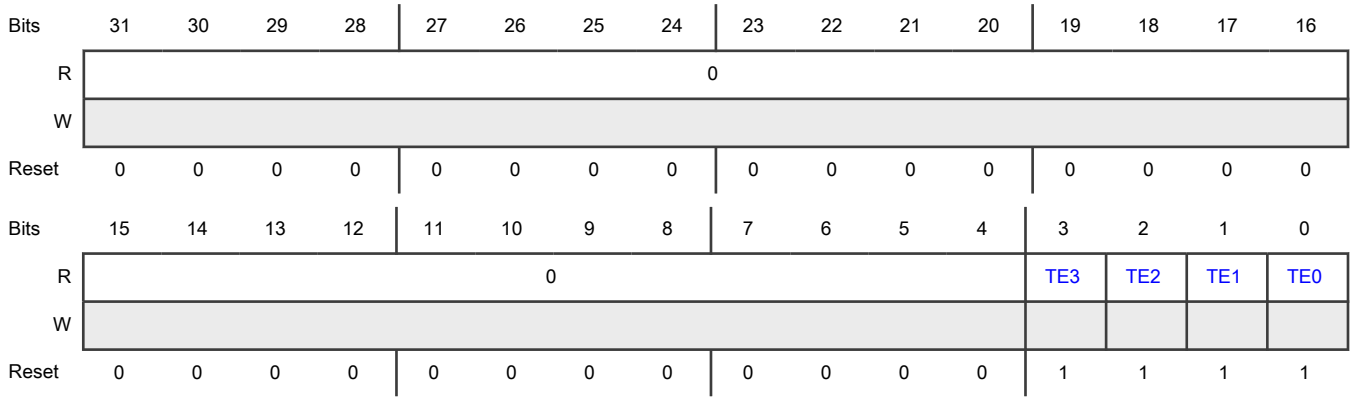
Offset

Register	Offset
TSR	124h

Function

The TSR register shows whether the MUB transmit registers are empty or not.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TEn	MUB Transmit Register n Empty • There are 4 Transmit Registers (n = 0 to 3) • The TEn bit is set to "1" after the MUA RRn register is read on the MUA side. • After the TEn bit is set to "1", it signals the MUB side that the MUB TRn register is ready to be written on the MUB side, and a Transmit n interrupt is issued on the MUB side (if the MUB_TCR[TIEn] bit is set to "1"). • TEn bit is cleared after the MUB TRn register is written on the MUB side. • After TEn bit is cleared, the Transmit n interrupt(if the MUB_TCR[TIEn] bit is set to "1") is cleared on the MUB side. • TEn bit is set to "1" when the MU resets. 0b - MUB TRn register is not empty. 1b - MUB TRn register is empty.

39.6.2.15 Receive Control Register (RCR)

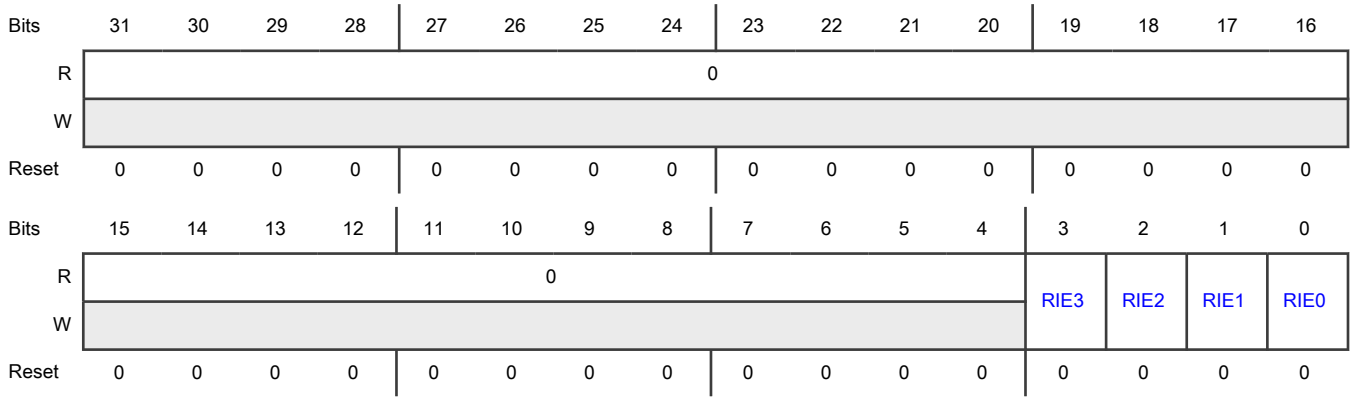
Offset

Register	Offset
RCR	128h

Function

The RCR register contains the MUB receive interrupt enables.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 RIEn	MUB Receive Interrupt Enable n • There are 4 Transmit Interrupts (n = 0 to 3) • If RIEn bit is set to "1", then an MUB Receive Interrupt n request is issued when the MUB_RSR[RFn] bit is set to "1". • If RIEn bit is cleared, then the value of the RFn bit is ignored and no MUB Receive Interrupt n request is issued. • RIEn bit is cleared when the MU resets. 0b - Disables MUB Receive Interrupt n. 1b - Enables MUB Receive Interrupt n.

39.6.2.16 Receive Status Register (RSR)

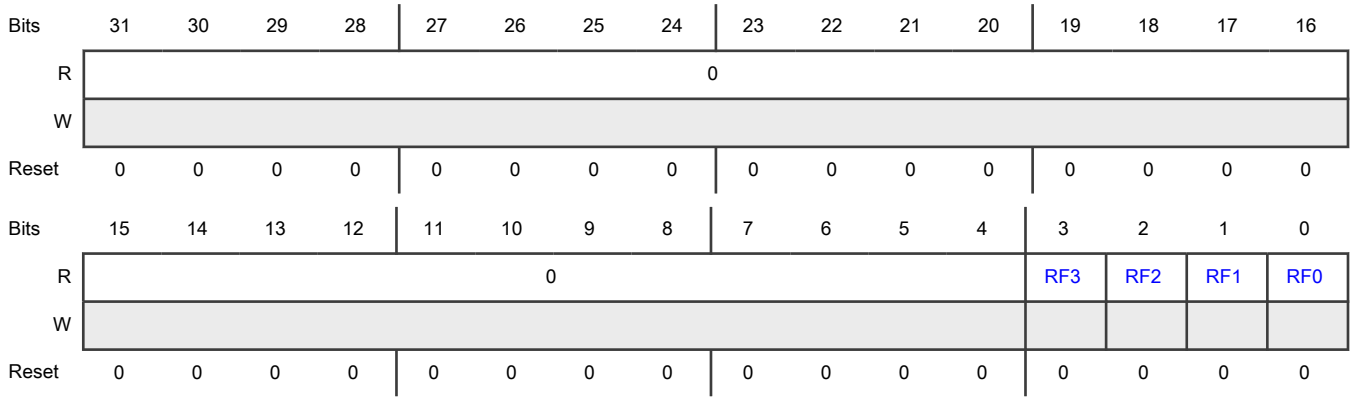
Offset

Register	Offset
RSR	12Ch

Function

The RSR register shows whether the MUB receive registers are full or not.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 RFn	MUB Receive Register n Full • There are 4 Receive Registers (n = 0 to 3) • The RFn bit is set to "1" when the MUA TRn register is written on the MUA side. • After the RFn bit is set to "1", the RFn bit signals the MUB side that new data in the MUB RRn register is ready to be read by the MUB, and a Receive n interrupt is issued on the MUB side (if the MUB_RCR[RIEn] has been set to "1"). • RFn bit is cleared when the MUB RRn register is read, or when the MU is reset. • After RFn bit is cleared, the Receive n interrupt(if the MUB_RCR[RIEn] has been set to "1") is cleared on the MUB side. 0b - MUB RRn register is not full. 1b - MUB RRn register has received data from MUA TRn register and is ready to be read by the MUB.

39.6.2.17 Transmit Register (TR0 - TR3)

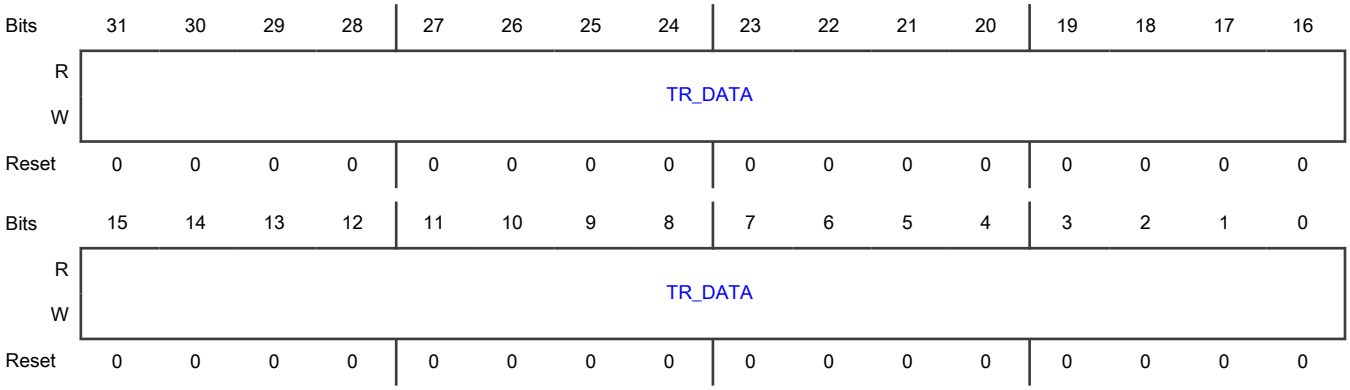
Offset

Register	Offset
TR0	200h
TR1	204h
TR2	208h
TR3	20Ch

Function

MUB Transmit Data.

Diagram



Fields

Field	Function
31-0 TR_DATA	MUB Transmit Data • Data written to the TRn register is reflected in the MUA Receive Register n (RRn). The TRn and RRn registers are not double-buffered, a write to the TRn register overrides the data readable at the RRn register. • A write to the transmit register clears the MUB_TSR[TEn] bit on the transmitter side, and sets the MUA_RSR[RFn] bit on the receiver side. • TRn register can be written only when the MUB_TSR[TEn] bit is set to "1". • Reading the TRn register returns all zeros.

39.6.2.18 Receive Register (RR0 - RR3)

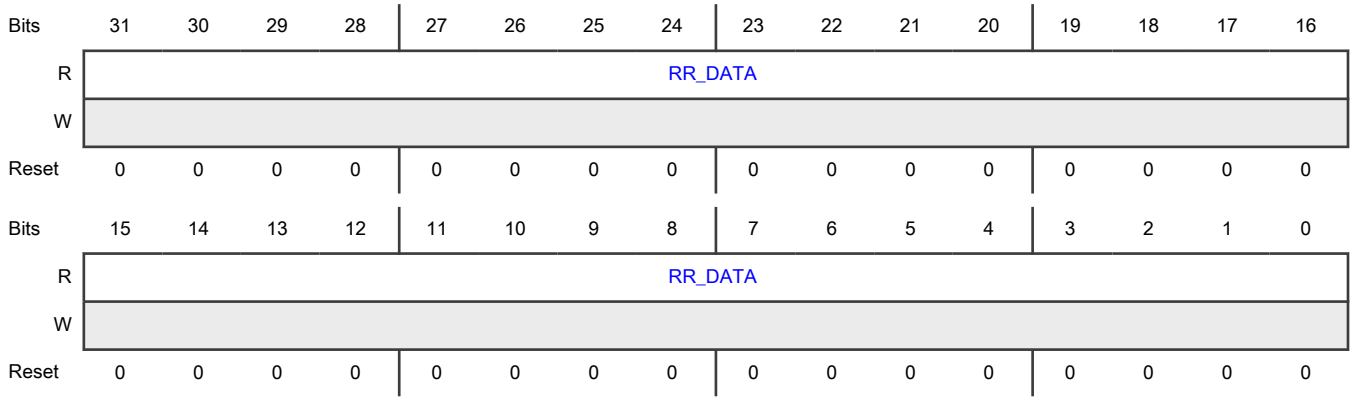
Offset

Register	Offset
RR0	280h
RR1	284h
RR2	288h
RR3	28Ch

Function

MUB Receive Data.

Diagram



Fields

Field	Function
31-0 RR_DATA	MUB Receive Data • Reflects the data written to MUA Transmit Register (TRn). • Reading the RRn register clears the MUB_RSR[RFn] bit on the receiver side, and sets the MUA_TSR[TEn] bit on the transmitter side. • RRn register can be read only when the MUA_RSR[RFn] bit is set to "1". Reading before MUA_RSR[RFn]=1 may result in reading incorrect data. Therefore, polling the MUA_RSR[RFn] bit to confirm it is set to "1" before reading RRn is required. • Writing to the RRn register generates an error response to the MUB.

39.7 Glossary

EP	Event Pending
GIR	General purpose Interrupt Request
GIP	General purpose Interrupt Pending
MUR	Messaging Unit Reset
NMI	Non-Maskable Interrupt
RF	Receiver Full
RFP	Receive Full Pending
TE	Transmitter Empty
TEP	Transmit Empty Pending

Chapter 40
Power Management

40.1 Introduction

The power management system generates, monitors, and controls power supplies and related resets. This chapter describes the system's interaction with other peripherals.

The power management system includes the following modules:

Table 208. Power management system modules and their functions

Part	Function
MC_ME	Initiates entry into Low-Power mode
MC_PCU	Controls entry into and exit from Low-Power mode
PMC	Regulates and monitors power supplies of the Run and the Standby domains
MC_RGM	Ensures a clean state and Run domain sanity by controlling the reset sequence
FIRC, WKPU, and other chip peripherals	Controls Standby mode wake-up and operations in Run and Standby modes

40.1.1 Power management system for MWCT2D16S and MWCT2D17S

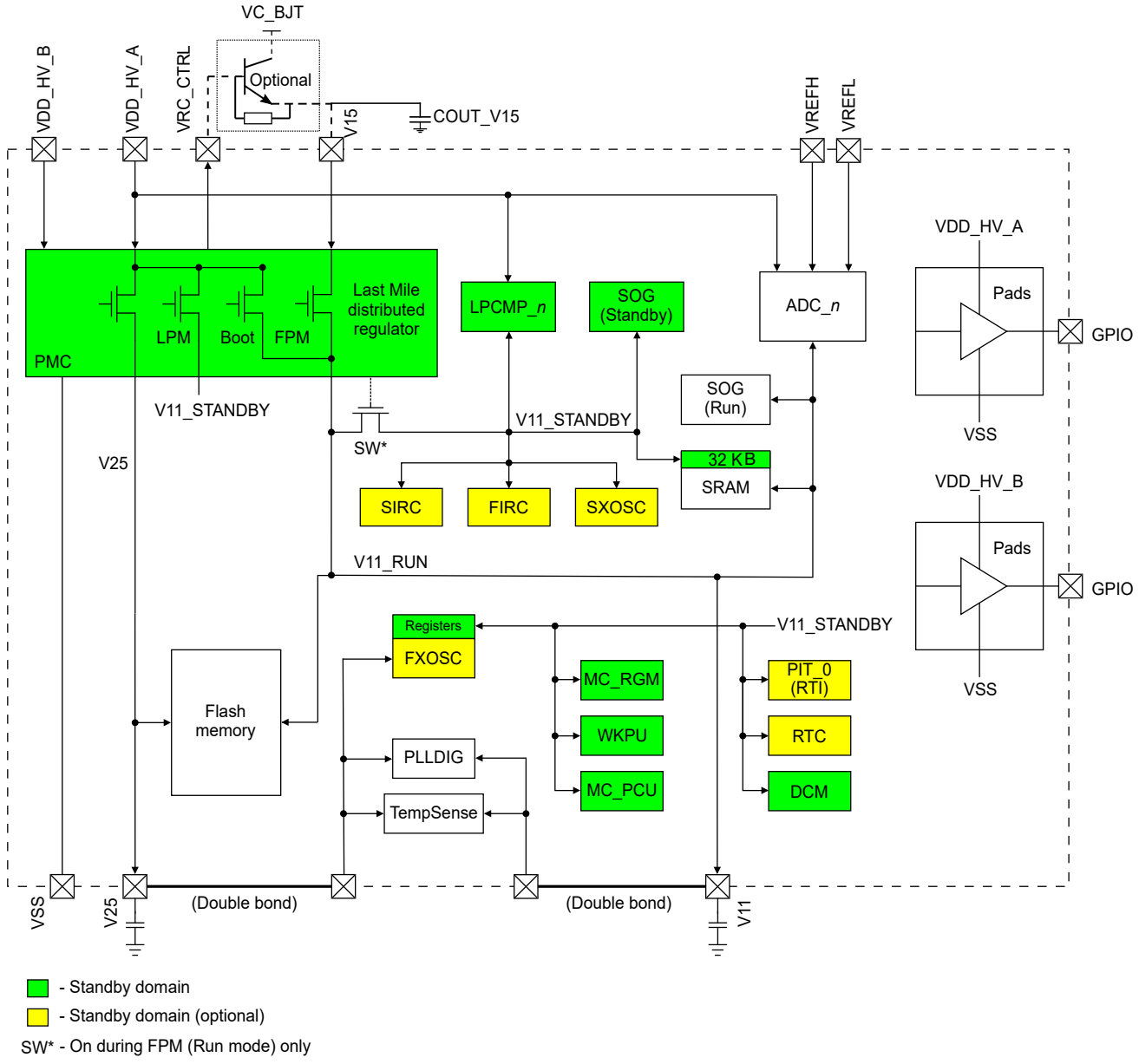


Figure 152. Power management system for MWCT2D16S and MWCT2D17S

40.1.2 Power management system for MWCT2016S and MWCT2015S

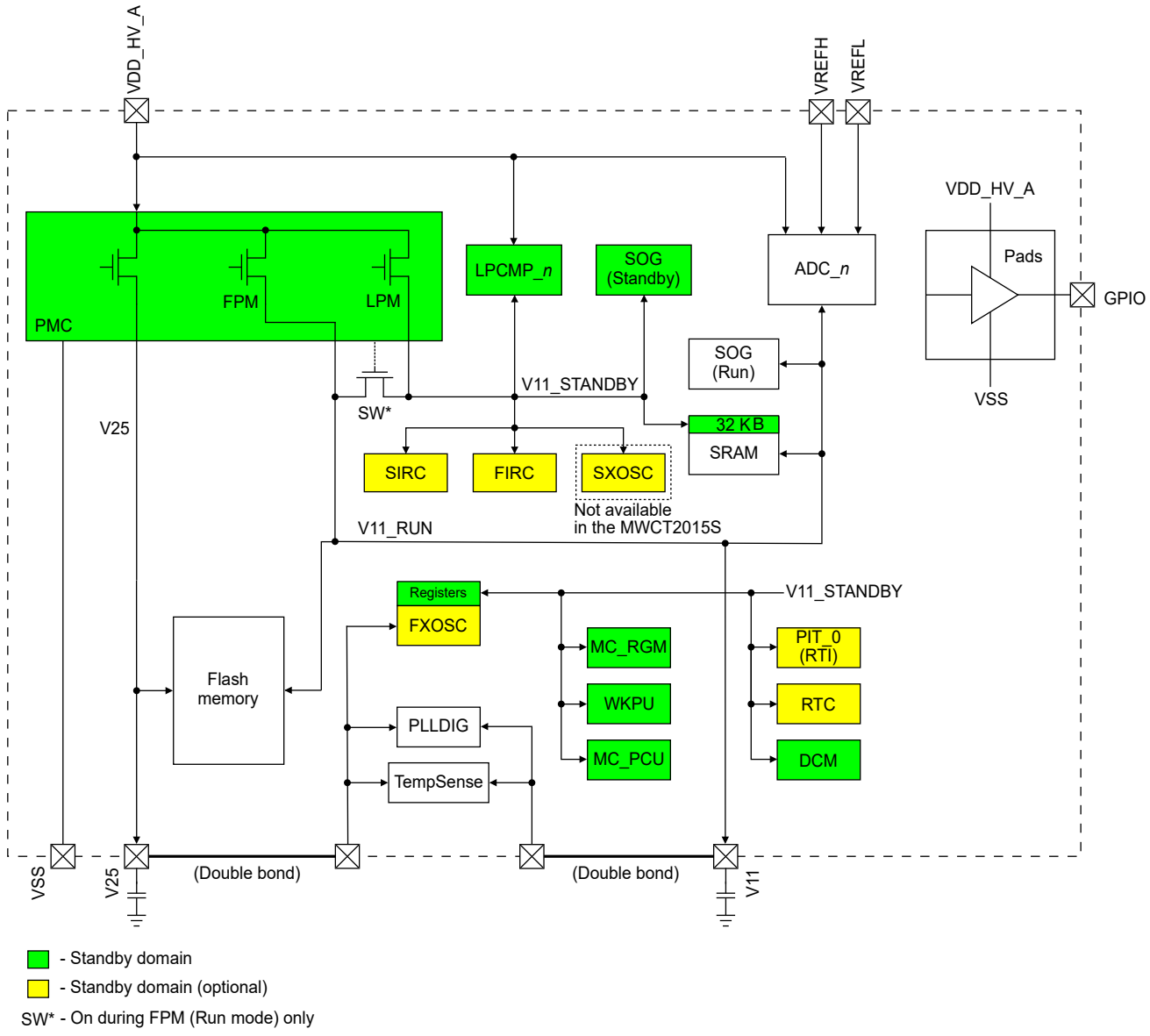


Figure 153. Power management system for MWCT2016S and MWCT2015S

40.1.3 Features

- Includes a linear regulator to use an optional external ballast (BJT) and VRC_CTRL output for 1.5 V generation. PMC uses 1.5 V supply and an internal regulator to generate 1.1 V (nominal) supply (V11_RUN) for the core logic
- Supports a low-power regulator (LPM) supplying core logic during Standby mode (V11_STANDBY)
- Includes two regulators for driving the logic in Run mode (both can be software-controlled):
  - Boot (Applicable for MWCT2D17S and MWCT2x16S)
  - Last-mile (FPM)
- Supports power switches to isolate voltage islands and configure the chip in Standby mode

- Includes a linear regulator that generates a 2.5 V supply (V25) from VDD_HV_A
- Supports voltage monitors ensuring transitions to a safe state (POR) when a supply is out of a valid range
- Controls power-mode transitions through an interaction with the digital interface
- Offers a padkeeping feature that retains I/O configuration after exiting Standby mode
- Separates ADC reference supplies (VREFH and VREFL)

40.1.4 Operational power modes

This chip uses two modes:

- Run mode (FPM): Main operation mode having full-chip performance and a higher current consumption as compared to Standby mode.
- Standby mode (LPM): Low-performance mode of the chip in which the Run domain is turned off. Most of the cores and peripherals turn off in this mode.

The boot regulator manages the chip during the booting process.

The last-mile regulator is the full-performance regulator, which you enable for running applications. The upcoming sections discuss the sequence to enable or disable the regulator.

The LPM regulator manages the chip in Standby mode.

40.1.5 External 1.5 V source

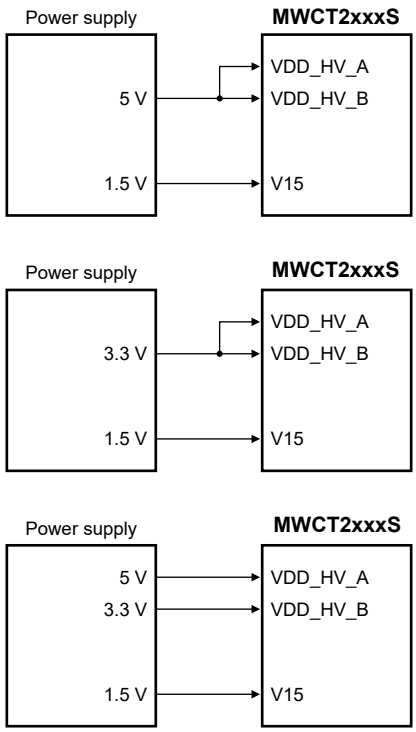


Figure 154. External 1.5 V source

40.1.6 Using a BJT for 1.5 V generation

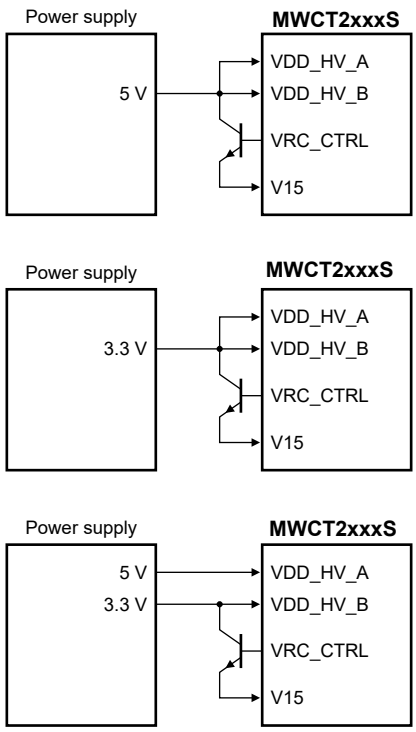


Figure 155. Using a BJT for 1.5 V generation

40.2 Power-up sequence

Figure 156 shows the reset sequence for a power-up or Standby mode event. This sequence starts from the POR phase. For the Run domain logic, Standby mode exit operation is same as power-up.

See the Reset Overview chapter for more information about the chip reset sequence.

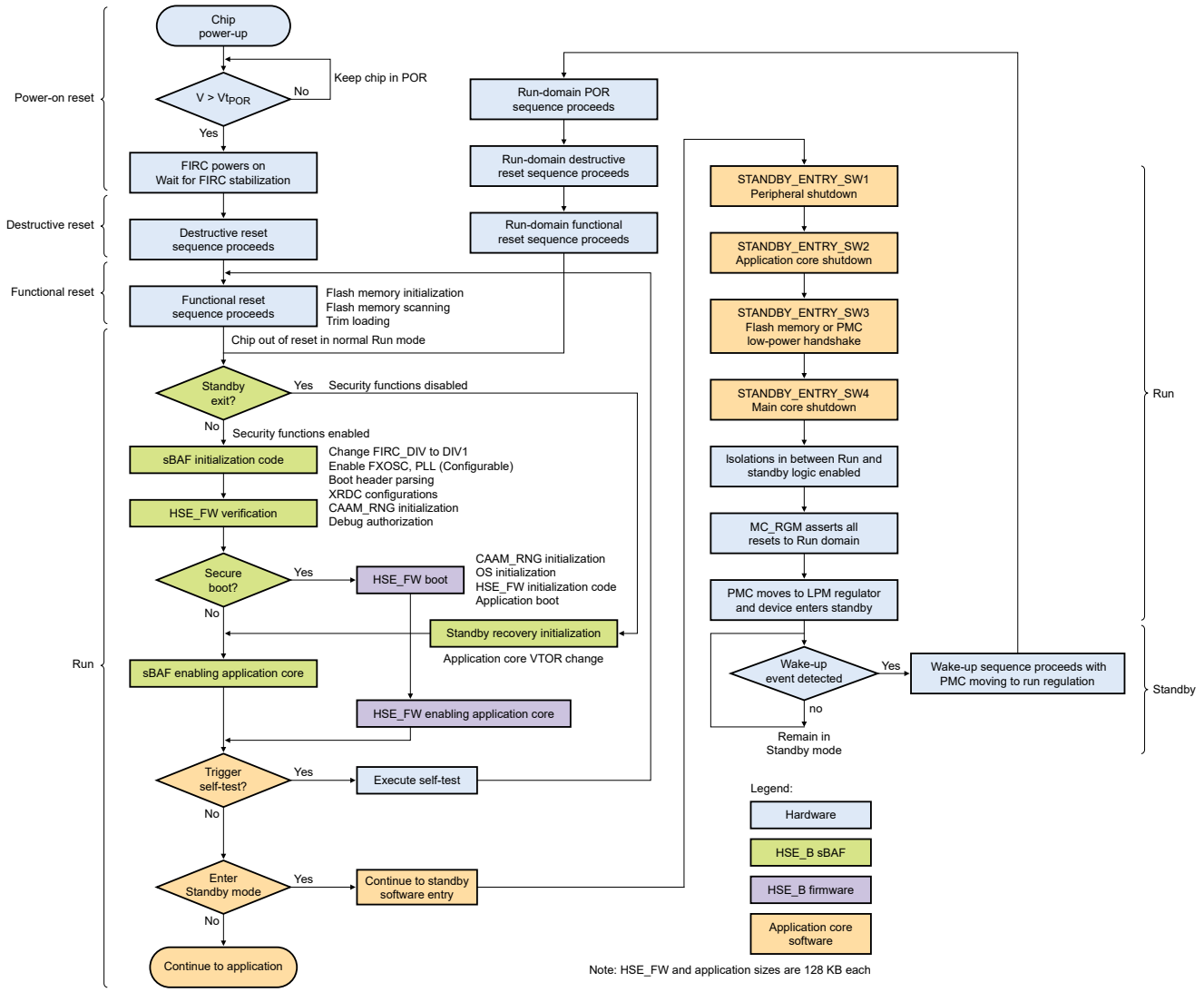


Figure 156. Power-up sequence

40.3 PMC last-mile regulator auto-enable feature

PMC includes an automatic last-mile auto-enable feature. After starting on boot regulator, this feature allows automatic switch over to last-mile regulator if 1.5 V is present during:

- Chip startup
- Standby mode recovery

You can control the PMC last-mile regulator by appropriately configuring:

- PMC's CONFIG[LMAUTOEN] field
- PMC's CONFIG[LM_EN] field

  For more information, see the descriptions of these fields in PMC's CONFIG register.

NOTE

You must write 1 to PMC's CONFIG[LM_EN] field before transtioning to faster clock frequencies irrespective of the setting of PMC's CONFIG[LMAUTOEN] field. This is because of the reduced clock speed when using the boot regulator.

40.3.1 Last-mile regulator with 1.5 V from an external source

Table 209. Last-mile regulator with 1.5 V from an external source

Operating condition	Last-mile regulator operation
After POR	Boots on boot regulator and then automatically switches to the last-mile regulator
After destructive reset	Remains on last-mile regulator **NOTE** If the destructive reset source is low voltage reset on 1.1 V then switchback happens from last-mile to boot regulator for boot.
After functional reset	Remains on last-mile regulator
After PMC's LVSC[LVD15S] field becomes 1	Switches, automatically, to the boot regulator to configure clocks for slow speed

40.3.2 Last-mile regulator using a BJT

Table 210. Last-mile regulator using a BJT

Operating condition	Last-mile regulator operation
After POR	Boots on boot regulator; switches to the last-mile regulator post reset when the Cortex-M7 core configures the software (post-secure boot) on FIRC (100 ms)
After destructive reset	Switches to boot regulator to check reset propagation delay
After functional reset	Remains on last-mile regulator
After PMC's LVSC[LVD15S] field becomes 1	Switches, automatically, to boot regulator to configure clocks for slow speed

40.4 Standby mode entry sequence

The Standby mode entry sequence includes three phases of operation:

- Standby mode entry configuration phase or software Standby mode entry sequence
- Standby mode entry handshake phase or hardware Standby mode entry sequence
- Standby mode entry or PMC Standby mode entry

NOTE

The Standby mode entry sequence described in this section is the only supported sequence. Contact NXP support if you require an alternate Standby mode entry sequence.

40.4.1 Software Standby mode entry sequence

The Standby mode entry sequence discusses chip configuration described in the sections that follow.

Before entering the software Standby mode sequence, the system clock source must be changed to FIRC at 48 MHz because PLLDIG is not available in Standby mode. In this mode, all clock sources can be optionally disabled (including FIRC, which results

in a no-clock, low-power consumption mode). You could use FXOSC, if enabled, when the 2.5 V supply is available by appropriate configuration of PMC's CONFIG[LPM25EN].

The software Standby mode entry sequence consists of four steps:

1. SW1: Module shutdown process
2. SW2: Application core shutdown process
3. SW3: Flash memory low-power handshake and PMC last-mile regulator control
4. SW4: Main core shutdown process

These processes are described in detail in the sections that follow. See Figure 156 that shows the relationship between these four steps in a flow diagram.

40.4.1.1 SW1: Module shutdown process

I/O and module configuration for Standby mode discusses the procedure to configure I/O and the chip modules for Standby mode.

The entry sequence for this mode includes module clock disabling steps (see the "Clocking" chapter for module clock turn on and turn off processes). You must use MC_ME's PRTN*n*_COFB*m*_CLKEN[REQ*p*] fields to enable or disable module clocks.

40.4.1.1.1 Disabling modules

Disable modules by configuring the appropriate fields in their registers for Standby mode operation. See specific module chapters for more information.

The Standby mode entry sequence includes the module clock disabling step, with which you can disable the modules that you do not need for Standby mode operation.

The sequence of disabling modules is shown in I/O and module configuration for Standby mode.

NOTE

While enabling or disabling the modules, you must verify that the module is disabled when you read MC_ME's PRTNn_COFBm_STAT register and the module disable field, if applicable. In case of a discrepancy, you must perform proper diagnostic steps.

You must clear the I/O controls for the pads that you do not require in Standby mode (OBE, IBE, and so on). This avoids any unwanted pad keeping settings. See Pad keeping for more information on the chip pad keeping process.

For any standby wake source, if an interrupt occurs it must be disabled before entering into standby and only the wake up event of that source must be enabled. This is to avoid any SW conflict in the interrupt handling for multi application core cases.

The MC_ME's PRTN*n*_COFB*m*_STAT register indicates the status of peripheral clock enable or disable. It may take up to three clock cycles for MC_ME's PRTN*n*_COFB*m*_STAT register to update after MC_ME's PRTN*n*_COFB*m*_CLKEN register is updated.

40.4.1.1.2 I/O and module configuration for Standby mode

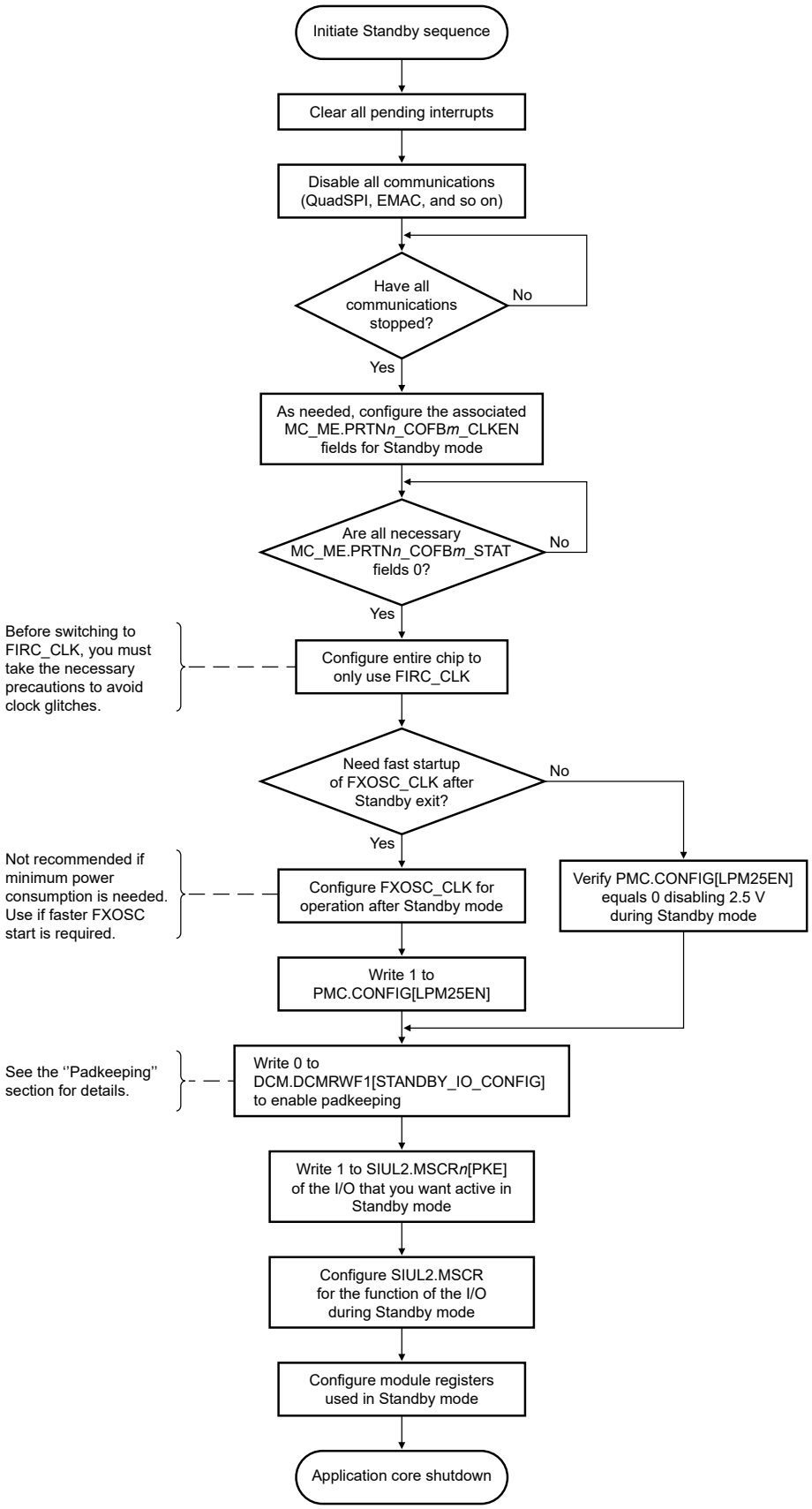


Figure 157. I/O and module configuration for Standby mode

40.4.1.2 SW2: Application core shutdown process

The application cores(s) execute a WFI (as opposed to the main core running the Standby mode entry sequence). See the section "Application core shutdown" in the MC_ME chapter for more information.

The main core configuration (programming valid core ID and enabling standby entry process) and wake-up source configuration must also be set in SW2, so that SW4 contains only the main core WFI execution.

40.4.1.3 SW3: Flash memory low-power handshake and PMC last-mile regulator

In this process, you execute a flash memory low-power handshake and disable the PMC last-mile regulator by executing the procedures indicated in Figure 158 and Figure 159 .

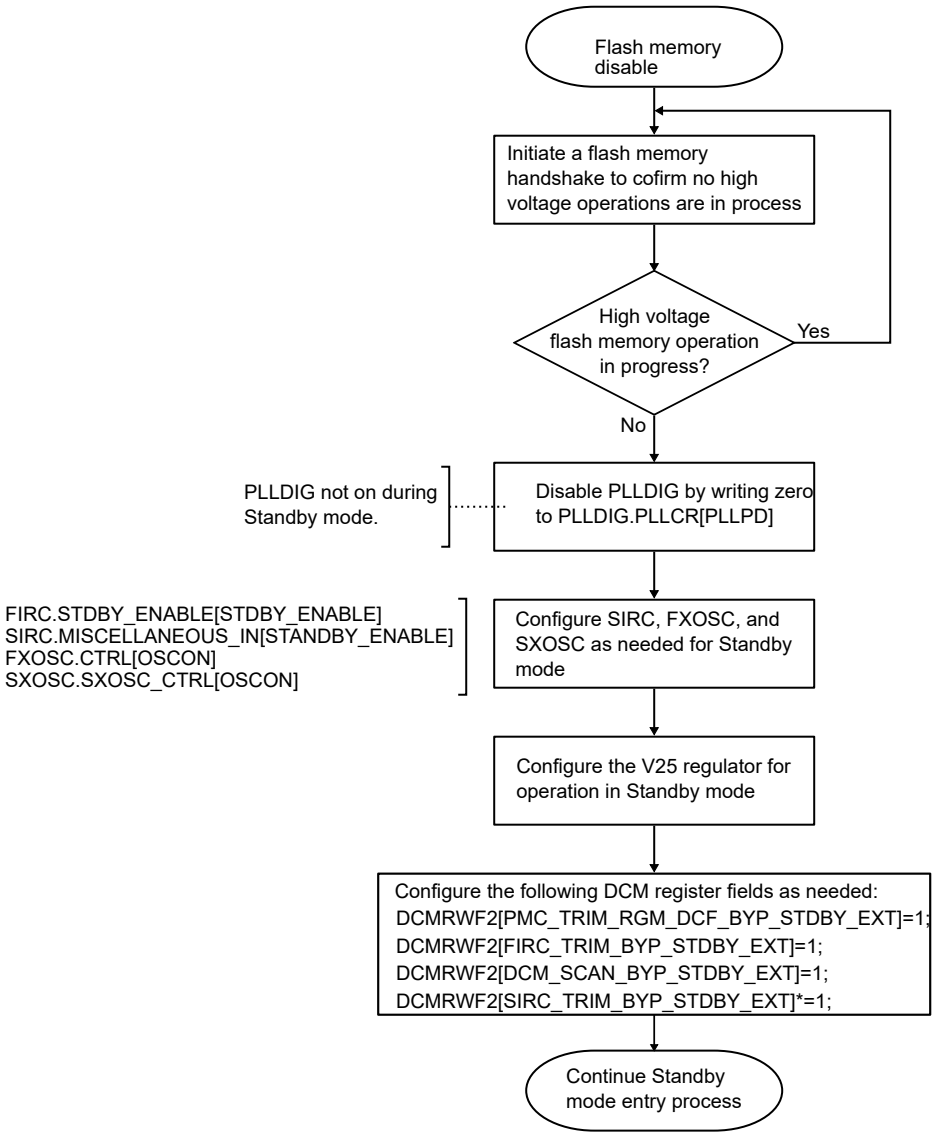


Figure 158. Flash memory Standby mode configuration

NOTE

Disable the last-mile regulator according to the Last-mile regulator disable sequence.

40.4.1.3.1 Last-mile regulator disable sequence

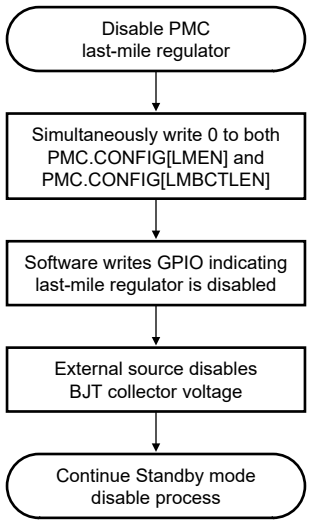


Figure 159. Last-mile regulator disable sequence

When exiting Standby mode, the 1.1 V capacitor is charging. You can charge the 1.5 V capacitor during or after the 1.1 V capacitor is charged. If you charge the 1.5 V capacitor sequentially after the 1.1 V capacitor, you will need additional time to complete the overall charging process.

Leaving PMCs CTRL[LMBCTLEN] = 1 saves time when exiting Standby mode and does not draw any additional current during this mode.

40.4.1.4 SW4: Main core shutdown process

For information on this process, see these in the "Mode Entry Module (MC_ME)" chapter:

- Figure "Standby entry sequence along with main core shutdown"
- Section "Main core shutdown and Standby mode entry"

You must configure WKPU before disabling interrupts to avoid missing any events as shown in the Standby entry sequence along with main core shutdown flowchart in the MC_ME chapter. You must program WKPU, disable interrupts, and configure MC_ME's Standby mode entry in SW1 (see SW1: Module shutdown process for more information). In SW4, you must perform only the main-core WFI execution.

40.4.2 Hardware Standby mode entry sequence

The hardware Standby mode entry sequence consists of handshaking between MC_PCU and MC_ME occuring after Software Standby mode entry sequence completes. The FSM in MC_PCU performs these steps automatically and does not require your intervention (see Power sequence FSM in the Power Control Unit (MC_PCU) chapter for more information).

40.4.2.1 PMC Standby mode entry

The PMC Standby mode entry sequence starts after Hardware Standby mode entry sequence completes, and consists of these phases:

1. Standby mode entry acknowledgment and initiation of an internal low-power process on receiving Standby mode entry request
2. Disabling of the boot regulator within the PMC low-power process. This point is only applicable for MWCT2D17S and MWCT2x16S.
3. Disabling of the V25 regulator and the FPM LVR monitors

4. Disabling of the V25 regulator (oscillator and flash memory supply) and the LPM monitor, unless it is enabled during Standby mode (see PMC's CONFIG[LPM25EN] field in the "Power management controller (PMC)" chapter for more information)
5. Disabling of the VDD_HV_B LPM monitors

40.5 Chip status at the end of Standby mode entry sequence

After PMC Standby mode entry completes, the chip completes Standby mode entry as follows:

1. Configures Standby domain peripherals according to SW1 (see SW1: Module shutdown process).
2. Enables pad keeping on pins as described in SW1 (see SW1: Module shutdown process). See Pad keeping for other details.
3. Powers down all memory types except Standby RAM (in SRAM0).
   - The V25 regulator can remain on during Standby mode. However, for maximum power savings, it must remain off in this mode.
4. Isolates Standby and Run domains from each other:
   - Standby domain is functional.
   - Run domain is held in reset.
5. Configures the system clock (FIRC_CLK or PLL_PHI*n*_CLK, depending on their configuration) and other clock sources according to SW3 (see Figure 158 for the procedure details).

   The cores are off and in the Standby domain.
6. Turns off the boot and FPM regulators.
7. Waits for a wake-up event to initiate recovery from Standby mode.

40.6 Chip operation in Standby mode

This chip supports the following functionalities in Standby mode:

- STANDBY_RAM content retention during Standby mode.
- Wake-up from up to 60 digital inputs (for details, see the signals WKUP[*n*] functions of WKPU module in the IOMUX file attached to this document). The section WKPU configuration on the chip in the "Wakeup Unit (WKPU)" chapter shows mapping of the wake-up sources.
- Wake-up from up to 16 analog inputs through the Trigger mode functionality (see the signals CMP*n*_IN*m* functions of CMP*n* modules in the IOMUX file attached to this document).
- Wake-up from on-chip timers:
  - RTI (function of PIT[0])
  - SWT0
  - RTC
- Ability to configure the chip clocking modules to optionally enable or disable in Standby mode (FIRC, SXOSC, FXOSC, and SIRC).

40.7 Standby mode exit

This chip supports Standby mode exit from a wake-up, functional reset, or destructive reset event. The sources that cause chip Standby mode exit are:

- MC_RGM functional reset event

- MC_RGM destructive reset event
- WKPU wake-up events, WKPU[0]–WKPU[63]. See the WKPU chapter for more information.

After Standby mode exit, the following events occur (for more information, see "Power sequence FSM" in the MC_PCU chapter for MC_PCU FSM transitions during entry into and exit from Standby mode):

1. A wake-up event arrives.
2. FIRC starts powering up (if disabled in Standby mode).
3. PMC starts the transition process to FPM (for example, enables the last-mile regulator and provides V11_RUN supply to the chip).
4. MC_PCU removes the isolation between Run and Standby domains.
5. Run domain reset deasserts (asserts on Standby mode entry) and the chip undergoes a functional reset exit sequence for this domain.
6. The chip enters Run mode of operation.

40.7.1 Faster Standby mode exit

The chip supports an optional configuration for faster recovery from Standby mode on the expense of a higher capacitor recharging current. See the CONFIG[FASTREC] field description in the PMC chapter for more information on faster PMC recovery from Standby mode.

This chip supports an optional feature that bypasses SIRC trimming, FIRC trimming, PMC trimming and DCM scanning, SIRC trimming is optionally bypassed or not, phases in case of Standby mode exit sequence by writing 1 to DCM's DCMRWF2[6], DCMRWF2[5], DCMRWF2[4], and DCMRWF2[3] fields respectively before Standby mode entry for bypass operation. Even if the FIRC trimming is bypass before entering to standbymode FIRC must be at 48MHz, this results in a considerable reduction in Standby mode exit duration. The trim values are retained across Standby mode and bypassing these values does not cause any impact.

Configure to achieve the tMODE_STDBYEXIT_FAST = 53us

DCMRWF2[PMC_TRIM_RGM_DCF__BYP_STDBY_EXT] = 1

DCMRWF2[FIRC_TRIM_BYP_STDBY_EXT] = 1

DCMRWF2[DCM_SCAN_BYP_STDBY_EXT] = 1

DCMRWF2[SIRC_TRIM_BYP_STDBY_EXT] = x

40.7.2 Last-mile regulator enable sequence

After Standby mode exit, you must re-enable the PMC last-mile regulator before transitioning to faster clock frequencies. Figure 160 shows the last-mile regulator enable sequence part of the Standby mode exit process.

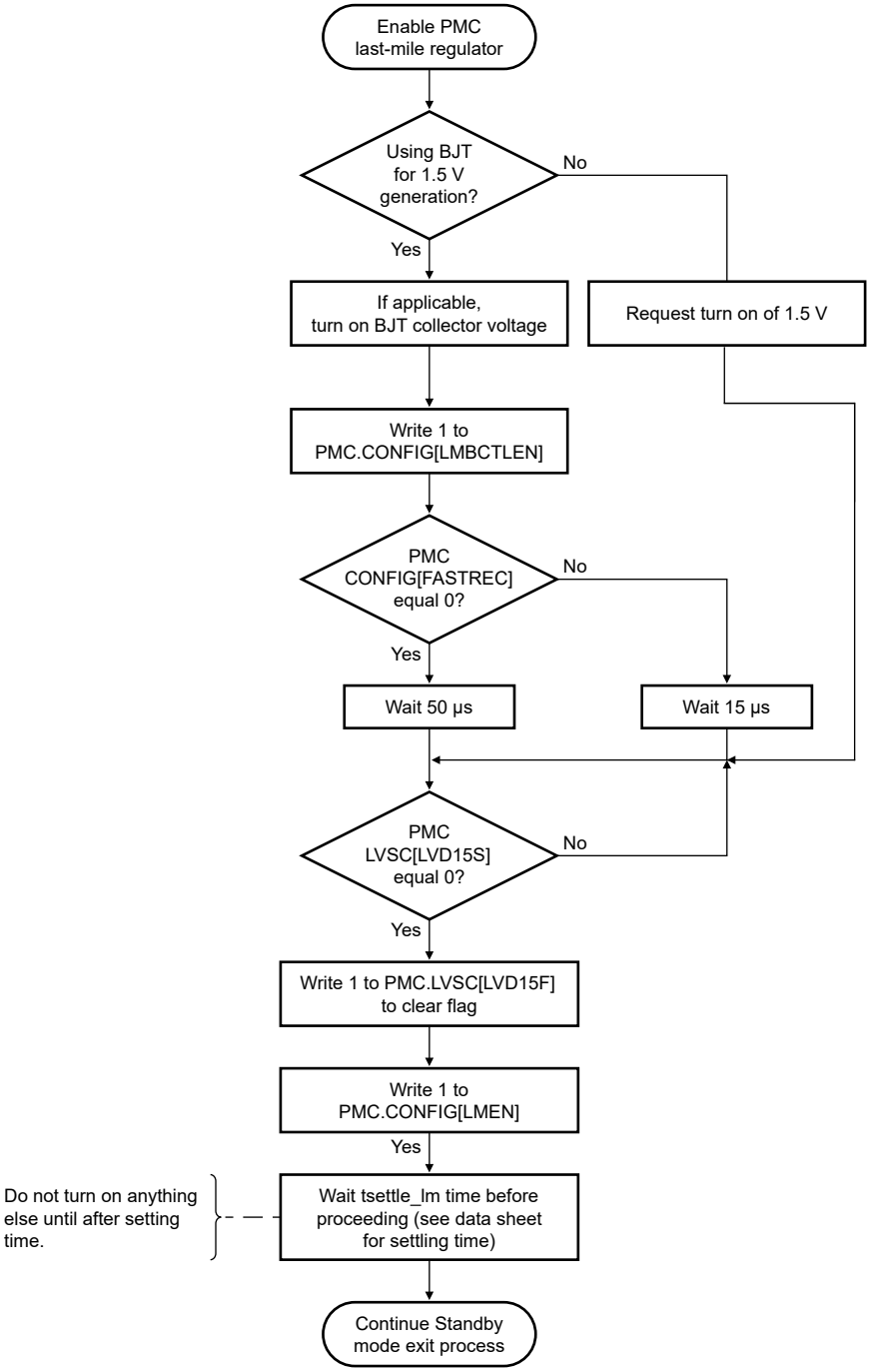


Figure 160. Last-mile regulator enable sequence

40.8 Chip power domain partitioning

These are the modules in the Standby domain. For more information, see "Peripheral reset status" in the "Reset Overview" chapter.

- FIRC
- FXOSC
- MC_RGM

- DCM
- WKPU
- PIT_0 (RTI)
- LPCMP_[0:2]
- PMC
- SIRC
- SXOSC
- RTC
- SWT_0
- DCF clients

40.8.1 Module power status

Table 211. Module power status

PD0 contents	MWCT2D17S	MWCT2D16S	MWCT2016S	MWCT2015S
External Pin wakeups	Minimum 60	Minimum 60	Minimum 60	33 on 100MQFP & 12 on 48LQFP
RTC_API	Yes	Yes	Yes	Yes
LPCMP0	Yes	Yes	Yes	Yes
LPCMP1	Yes	Yes	Yes	No
LPCMP2	Yes	No	No	No
SWT0	Yes	Yes	Yes	Yes
WKPU	Yes	Yes	Yes	Yes
PMC	Yes	Yes	Yes	Yes
RGM	Yes	Yes	Yes	Yes
SXOSC(only used for RTC)	Yes	Yes	Yes	No
FXOSC	Yes	Yes	Yes	Yes
FIRC	Yes	Yes	Yes	Yes
SIRC	Yes	Yes	Yes	Yes
CLK OUT	Yes	Yes	Yes	Yes
SRAM	32K	32K	32K	32K
PIT_RTI_0	Yes	Yes	Yes	Yes
DCM and DCF records	Yes	Yes	Yes	Yes
DCM Flash Interface	Yes	Yes	Yes	Yes

SWT0 (Cortex-M7_0) resides in the Standby domain and supports a configurable hardware-based timer operation during Standby mode depending on the configuration of SWTs.

All clock sources (excpet PLLDIG) are available in Standby mode. SIRC is present in the Standby domain for low-power wake-up, but can be enabled.

NOTE

This chip does not support Stop and Wait modes. The only low-power mode it supports is Standby.

40.9 Pad keeping

Pad keeping allows you to return to I/O Run mode configuration settings after using I/O during Standby mode. It eliminates I/O reconfiguration in Run mode.

40.9.1 Pad keeping on Run domain pins

The process of entering Standby mode ensures that the chip maintains the Run domain configuration settings. Software Standby mode entry sequence specifies that you must write 0 to DCM's DCMRWF1[STANDBY_IO_CONFIG] before configuring I/O pad keeping. You could consider DCM's DCMRWF1[STANDBY_IO_CONFIG] field as a global enable for all pad keeping purposes during Standby mode. If you are unable to write to this field as described, pad keeping works as explained in this chapter, during Standby mode.

After the chip exits Standby mode, you must write 1 to DCM's DCMRWF1[STANDBY_IO_CONFIG] field (see "Chip specific register descriptions" in the DCM chapter). Writing 1 to this field disables pad keeping and places chip I/O according to its Run mode configuration.

40.9.2 Pad keeping on Standby domain pins

The Standby domain pads can have pad keeping enabled or disabled based on pad availability in Standby mode. Writing to SIUL2's MSCR*n*[PKE] field configures Standby mode pad keeping on a selected I/O.

Ensure that DCM's DCMRWF1[STANDBY_IO_CONFIG] = 0 before Standby mode entry for pad keeping on Standby domain pads. Write 0 to the field in case its value is not 0 already.

If a pad is not required during Standby mode, the corresponding MSCR*n*[PKE] field in SIUL2 must be 0.

40.9.3 Pad keeping configuration procedure

This procedure specifies how to enable an I/O for pad keeping:

1. Configure SIUL2's MSCR*n* register to control the pad state prior to Standby mode entry (for example, writing 0 to SIUL2's MSCR*n*[OBE], MSCR*n*[IBE], and MSCR[PUE] fields tristates the corresponding I/O).
2. Configure SIUL2's MSCR*n*[PKE] field as needed for an I/O pad keeping state during Standby mode.
3. Write 0 to DCM's DCMRWF1[STANDBY_IO_CONFIG].

   The application core executes WFI, and Standby mode sequence starts.
4. Write 1 to DCM's DCMRWF1[STANDBY_IO_CONFIG] field on Standby mode exit to disable pad keeping. This configures SIUL2 according to its previous settings.

If you write 1 to DCM's DCMRWF1[STANDBY_IO_CONFIG] field before entering Standby mode, the isolation removal hardware removes pad keeping on Standby mode exit. This (writing 1 to DCM's DCMRWF1[STANDBY_IO_CONFIG] field before Standby mode entry) is useful in case of low-power debug because enabling pad keeping does not allow low-power debug protocol to work properly (because the TDO pad is padkept). For low-power debug, you must write 1 to DCM's DCMRWF1[STANDBY_IO_CONFIG] field prior to Standby mode entry.

In case of Standby mode exit by reset, the pad keeping of the reset pin is removed when the chip is reset on Standby mode wake up. See the "GPIO padkeeping enable" signal in Figure 162. The signal corresponds to wake-up via reset event case.

40.9.4 Pad keeping waveforms

40.9.4.1 Pad keeping when the chip wakes up from Standby mode via an interrupt event

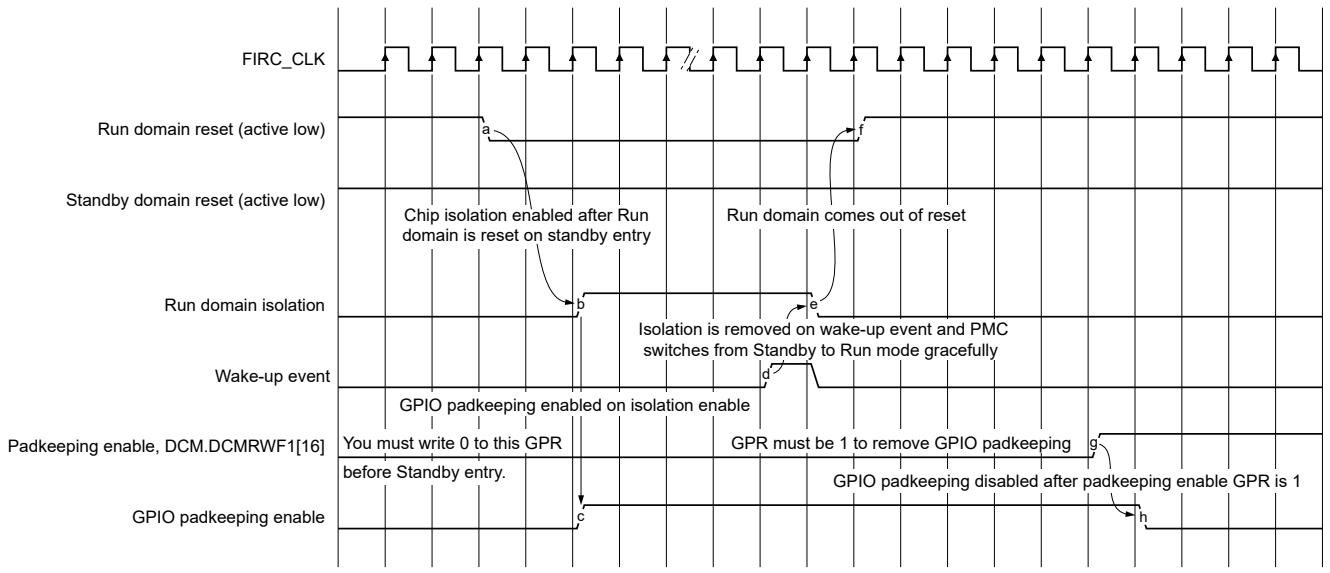


Figure 161. Pad keeping when the chip wakes up from Standby mode via an interrupt event

40.9.4.2 Pad keeping when the chip wakes up from Standby mode via reset

If the chip exits Standby mode via reset, the reset pad keeping is removed when the chip resets after Standby-mode wake-up. See the figure's "GPIO padkeeping enable" waveform corresponding to wake-up via a reset event.

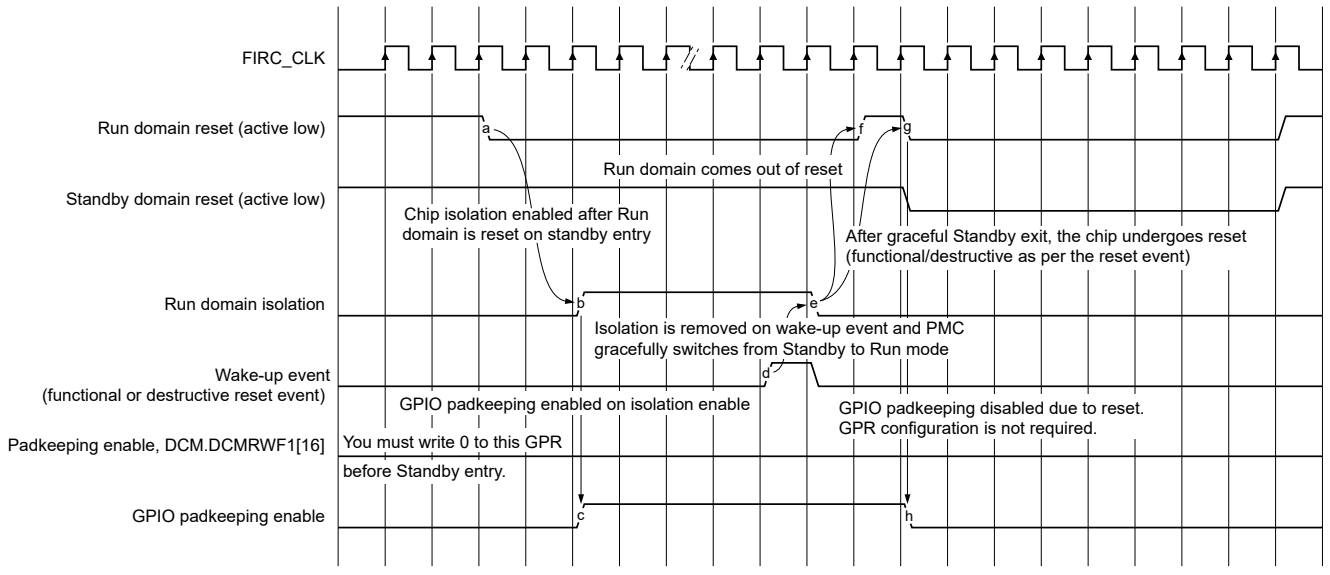


Figure 162. Pad keeping when the chip wakes up from Standby mode via reset

40.9.5 SIUL2's fields for actively-driven pins in Standby mode

If you use a WKPU pin as a wake-up input, perform these operations on SIUL2's fields:

1. Write 1 to MSCR*x*[IBE].

2. Program MSCR*x*[PUE] and MSCR*x*[PUS] according to the use case.
3. Write 0 to MSCR*x*[PKE].

For the GPIO pins that are driven to high impedance during Standby mode:

1. Write 0 to DCM's DCMRWF1[STANDBY_IO_CONFIG] field.
2. Write 0 to SIUL2's MSCR*x*[SSS] field (GPIO mode).
3. Write 0 to SIUL2's MSCR*x*[IBE] and MSCR*x*[OBE] fields.

Some of the pins are, by default, actively driven in Standby mode. If these pins retain a static value throughout the mode, program the corresponding MSCR*x*[PKE] bits individually.

This table shows the list of pins available in Standby mode along with the functions they perform.

Table 212. Active pins in Standby mode

Pin[1]	Function
GPIO_4	CMP0_OUT
GPIO_5	RESET_b
GPIO_9	CMP2_OUT
GPIO_11	CMP0_RRT
GPIO_12	CLKOUT_STANDBY; CMP1_OUT
GPIO_69	CMP2_RRT
GPIO_110	CMP0_RRT
GPIO_131	CMP0_OUT
GPIO_138	CLKOUT_STANDBY
GPIO_143	CMP1_RRT
GPIO_158	CMP1_OUT
GPIO_159	CMP1_RRT
GPIO_174	CMP0_OUT
GPIO_175	CMP0_RRT
GPIO_196	CMP2_OUT
GPIO_197	CMP2_RRT

1. See the IOMUX sheet attached to this document for details on pins configuration in different chips.

40.10 Glossary

Application core Core apart from the main core

Boot regulator The on chip 1.1 V regulator that is active during startup and entry and exit from Standby mode

FSM Finite state machine

FPM Full-power mode (Run mode)—uses the last-mile regulator and 1.5 V source to generate the 1.1 V core logic supply during a full-power operation (V11_RUN)

LPM Low-power mode (Standby mode)—uses the low-power regulator to generate the 1.1 V core logic supply during low-power operation (V11_STANDBY)

LVR	Low voltage reset
Main core	Core initiating the chip Standby mode request (for example, the core corresponding to the "core index" in MC_ME's MAIN_COREID register)
Pad keeping	Maintains I/O pad configuration during Standby mode, if enabled
V11_STANDBY	Core logic and clock sources, low-voltage supply to Standby domain
V11_RUN	Low-voltage supply to Run domain
V15	High-current input for core or logic supply from either an external BJT or from direct 1.5 V external supply
V25	Flash memory, FXOSC, and PLLDIG high-voltage supply
VDDA_ADC	ADC supply voltage
VREFH	ADC high-voltage reference supply
VREFL	ADC low-voltage reference supply
VSS	Core logic ground supply
VDD_HV_A	Main I/O voltage supply (5 V or 3.3 V)
VDD_HV_B	Other I/O domain voltage supply (5 V or 3.3 V)
VRC_CTRL	PMC voltage regulator control output using the BJT option to generate a 1.5 V supply
WFI	Wait for interrupt software instruction

Chapter 41
Power Management Controller (PMC for MWCT2D17S and MWCT2D16S)

41.1 Introduction

PMC is the power management controller for the MWCT2S family of microcontrollers. It provides multiple power options to allow you to optimize power consumption for the level of functionality needed. It includes internal voltage regulators, POR, and the integrated low/high voltage detect system with reset (brown-out) capability. The voltage regulator requires a 3.3 V or 5 V input to generate all the required secondary supplies.

41.2 Features

PMC includes the following features:

- Combination of internal and external voltage regulator options, offering RUN and Standby modes
- Active POR providing brown-out detect
- LVR for all system-relevant power domains
- LVD and HVD as indication for software

41.3 Modes of operation

PMC provides two basic modes of operation for the voltage regualtors and monitors:

- FPM, which is used on chip-level in RUN modes: For high-current consumption
- LPM, which is used on chip-level for Standby modes: For low-current consumption

41.4 Block diagram

The following figure shows the block diagram for this module.

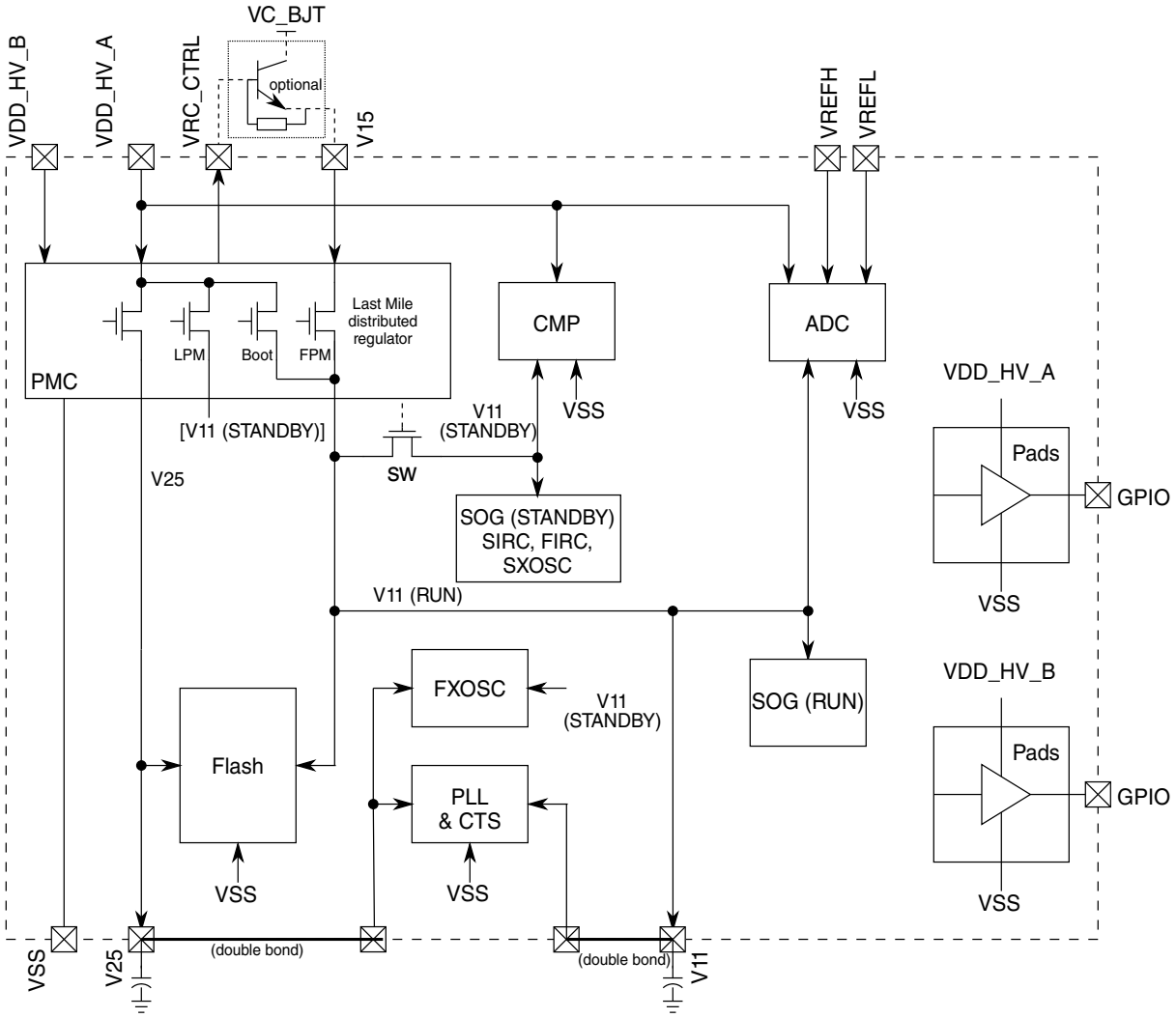


Figure 163. PMC block diagram

41.5 Signals

This table describes the PMC module signals.

Table 213. Signal Description

Signal	I/O	Description
VDD_HV_A	Supply input	This is the primary high-voltage supply input to PMC. VDD_HV_A is used for the PMC internal precision references. After the VDD_HV_A domain is powered up, it must be kept powered at all times of operation (FPM and LPM).
VDD_HV_B[1]	Supply input	This is the secondary high-voltage supply input supervised by the PMC. After the VDD_HV_B domain is powered up, it must be kept powered at all times of operation (FPM and LPM).

Table continues on the next page...

Table 213. Signal Description (continued)

Signal	I/O	Description
V25	Supply output	V25 power supply domain is driven by a fully integrated low-dropout linear voltage regulator. It supplies the Flash memory and (via a double bond) the clock modules.
V15[1]	Supply input	This is the high-current input for core/logic supply that can be fed by an external BJT or another source.
VRC_CTRL	Output	VRC_CTRL connects to the base of external BJT, if this option is used to generate V15.
V11	Supply output	V11 is the core/logic supply. It is driven by a fully integrated low-dropout linear voltage regulator.
VSS	Ground	VSS must be grounded. All VSS Pins need to be externally connected to the same ground node.

1. VDD_HV_B and V15 not present in MWCT2015S and MWCT2016S.

41.6 Functional description

The following sections describe functional details of the PMC.

41.6.1 Reset

The POR and all LVRs are combined into one single MCU POR.

After an MCU POR event, it can be determined which power domain caused it, by reading in the PMC_LVSC register, the POR flag, and LVR flags.

After an initial power ramp up of the MCU in the PMC_LVSC register, the POR flag and the LVR flags are all set to 1. The Go/Nogo flags have an arbitrary value.

NOTE

After an initial power ramp up, all flags in the LVSC register must be cleared (by writing 0xFFFFFFFF to the LVSC register).

Because the flags are sticky bits, it is required to clear them before usage. So, in case of an unxepected MCU POR, the source of the problem can be tracked and debugged by reading the flags in the LVSC register.

41.6.2 Interrupts

PMC includes two interrupt sources:

- HVD interrupt: It combines all HVD monitors into one interrupt source. Interrupt enable is the HVDIE field in the CONFIG register. See the PMC Configuration Register (CONFIG) and Low Voltage Status and Control Register (LVSC) registers for details.
- LVD interrupt: It combines the LVD15 and LVD5A monitors into one interrupt source. Interrupt enable is the LVDIE field in the CONFIG register. See the CONFIG and LVSC registers for details.

41.7 PMC register descriptions

41.7.1 PMC memory map

This section includes the PMC module memory map and detailed descriptions for all the registers.

PMC_2D17 base address: 0h

Offset	Register	Width (In bits)	Access	Reset value
0h	Low Voltage Status and Control Register (LVSC)	32	RW	See section
4h	PMC Configuration Register (CONFIG)	32	RW	See section
Ch	Version ID register (VERID)	32	R	0200_0001h

41.7.2 Low Voltage Status and Control Register (LVSC)

Offset

Register	Offset
LVSC	0h

Function

This register contains status and control bits to support the low-voltage reset and low- or high-voltage detect function. When the PMC is in LPM, the low- or high-voltage detect systems are disabled.

NOTE

For all flags that are not affected by reset (POR flag, all LVR flags, all GNG flags), in case a reset occurs at the same time while trying to clear the flags (by writing 1), the flag value is not defined appropriately. In this case, you need to clear the flag again after exit from reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PORF	0					GNG11OS...	GNG25OS...	LVR11LPF	LVR11F	LVR25LPF	LVR25F	LVRBLPF	LVRBF	LVRALPF	LVRAF
W	W1C						W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	u	0	0	0	0	0	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		LVD15S	LVD5AS	HVD11S	HVD25S	HVDBS	HVDAS	0		LVD15F	LVD5AF	HVD11F	HVD25F	HVDBF	HVDAF
W											W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PORF	POR flag Indicates that a power-on reset event has occurred. Writing 1 to this field clears it, and other reset sources have no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No power-on reset event has occurred 1b - Power-on reset event has occurred
30-26 —	Reserved
25 GNG11OSCF	Go/NoGo detect flag on Osc part of V11 domain Indicates that the Go/NoGo sensor has detected a low voltage on the V11 domain in FPM. This applies to only that part of the domain which supplies the 1.1V clocking modules (for example, PLL). Writing 1 to the field clears it, and other reset sources have no effect. 0b - No event has occurred 1b - NoGo event has occurred
24 GNG25OSCF	GO/NoGo detect flag on Osc part of V25 domain Indicates that the Go/NoGo sensor has detected a low voltage on the V25 domain in FPM. This applies to only that part of the domain which supplies the 2.5V clocking modules (for example, XOSC and IRC). Writing 1 to the field clears it, and other reset sources have no effect. 0b - No event has occurred 1b - NoGo event has occurred
23 LVR11LPF	LVR11LP flag on V11 domain Indicates that a low-voltage reset event has occurred on the 1.1V V11 power domain (FPM or LPM). Writing 1 to the field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
22 LVR11F	LVR11 flag on V11 domain in FPM Indicates that a low-voltage reset event has occurred on the 1.1V V11 power domain in the FPM. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
21 LVR25LPF	LVR25LP flag on V25 domain Indicates that a low-voltage reset event has occurred on the V25 power domain (FPM or LPM). Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
20 LVR25F	LVR25 flag on V25 domain in FPM Indicates that a low-voltage reset event has occurred on the V25 power domain in FPM. Writing 1 to this field clears it, and other reset sources have no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
19 LVRBLPF	LVRBLP flag on VDD_HV_B domain Indicates that a low-voltage reset event has occurred on the VDD_HV_B power domain (FPM or LPM). Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
18 LVRBF	LVRB flag on VDD_HV_B domain in FPM Indicates that a low-voltage reset event has occurred on the VDD_HV_B power domain in FPM. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
17 LVRALPF	LVRALP flag on VDD_HV_A domain Indicates that a low-voltage reset event has occurred on the VDD_HV_A power domain (FPM or LPM). Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
16 LVRAF	LVRA flag on VDD_HV_A domain in FPM Indicates that a low-voltage reset event has occurred on the VDD_HV_A power domain in FPM. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
15-14 —	Reserved
13 LVD15S	LVD15 status on V15 domain in FPM Shows the status of the 1.5V low-voltage detect, LVD15, on the V15 power domain. This monitor indicates when the V15 voltage level generated from external is on target. This feature is available only in FPM and disabled in LPM. After a reset or wakeup from LPM, the software should clear the LVD15F flag and check the status bit LVD15S to determine voltage level on V15 supply. 0b - Voltage on V15 is above low-voltage detect threshold or LPM. 1b - Voltage on V15 is below low-voltage detect threshold and FPM.
12 LVD5AS	LVD5A status on VDD_HV_A domain in FPM Shows the status of the 5V low-voltage detect, LVD5A, on the VDD_HV_A power domain. This monitor indicates if the voltage is below a certain threshold, which is set slightly below 4.5V (see Datasheet for

Table continues on the next page...

Table continued from the previous page...

Field	Function
	exact value). The feature is only available in FPM and disabled in LPM. After a reset or wakeup from LPM, the software should clear the LVD5AF flag and check the status bit LVD5AS to determine voltage level on VDD_HV_A supply. 0b - Voltage on VDD_HV_A is above low-voltage detect threshold 1b - Voltage on VDD_HV_A is below low-voltage detect threshold
11 HVD11S	HVD11 status on V11 domain in FPM Shows the status of the high-voltage detect, HVD11, on the V11 power domain. This feature is only available in FPM and disabled in LPM. 0b - Voltage on V11 is below high-voltage detect threshold or LPM. 1b - Voltage on V11 is above high-voltage detect threshold and FPM.
10 HVD25S	HVD25 status on V25 domain in FPM Shows the status of the high-voltage detect, HVD25, on the V25 power domain. The feature is only available in FPM and disabled in LPM. 0b - Voltage on V25 is below high-voltage detect threshold or LPM. 1b - Voltage on V25 is above high-voltage detect threshold and FPM.
9 HVDBS	HVDB status on VDD_HV_B domain in FPM Shows the status of the high-voltage detect, HVDB, on the VDD_HV_B power domain. The feature is only available in FPM and disabled in LPM. 0b - Voltage on VDD_HV_B is below high-voltage detect threshold or LPM. 1b - Voltage on VDD_HV_B is above high-voltage detect threshold and FPM.
8 HVDAS	HVDA status on VDD_HV_A domain in FPM Shows the status of the high-voltage detect HVDA on the VDD_HV_A power domain. The feature is only available in FPM and disabled in LPM. 0b - Voltage on VDD_HV_A is below high-voltage detect threshold or LPM. 1b - Voltage on VDD_HV_A is above high-voltage detect threshold and FPM.
7-6 —	Reserved
5 LVD15F	LVD15 flag on V15 domain in FPM PMC writes 1 to the LVD15F field when the LVD15S status field changes. To clear LVD15F, write 1 to it. If enabled, LVD15F causes an interrupt request. 0b - LVD15S has not changed. 1b - LVD15S has changed.
4	LVD5A flag on VDD_HV_A domain in FPM

Table continues on the next page...

Table continued from the previous page...

Field	Function
LVD5AF	PMC writes 1 to this field when LVD5AS status field changes. To clear LVD5AF, write 1 to it. If enabled, LVD5AF causes an interrupt request. 0b - LVD5AS has not changed. 1b - LVD5AS has changed.
3 HVD11F	HVD11 flag on V11 domain in FPM PMC writes 1 to this field when the HVD11S status field changes. To clear HVD11F, write 1 to it. If enabled, HVD11F causes an interrupt request. 0b - HVD11S has not changed. 1b - HVD11S has changed.
2 HVD25F	HVD25 flag on V25 domain in FPM PMC writes 1 to the HVD25F field when HVD25S status field changes. To clear HVD25F, write 1 to it. If enabled, HVD25F causes an interrupt request. 0b - HVD25S has not changed. 1b - HVD25S has changed.
1 HVDBF	HVDB flag on VDD_HV_B domain in FPM PMC writes 1 to the HVDBF field when HVDBS status field changes. To clear HVDBF, write 1 to it. If enabled, HVDBF causes an interrupt request. 0b - HVDBS has not changed. 1b - HVDBS has changed.
0 HVDAF	HVDA flag on VDD_HV_A domain in FPM PMC writes 1 to the HVDAF field when HVDAS status field changes. To clear HVDAF, write 1 to it. If enabled, HVDAF causes an interrupt request. 0b - HVDAS has not changed. 1b - HVDAS has changed.

41.7.3 PMC Configuration Register (CONFIG)

Offset

Register	Offset
CONFIG	4h

Function

If the Last Mile Regulator option is not available on your device (Version ID register (VERID) register: LMFEAT=0), then the bits LMEN, LMBCTLEN, LMAUTOEN and LMSTAT fields can not be written and read always 0.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						LVDIE	HVDIE	0			LVRBLPEN	LPM25EN	FASTREC	LMBCTLEN	LMEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved
17 LMSTAT	Last Mile regulator status bit This bits reflects the current status of the Last Mile regulator. This information is required as when auto turn over feature is enabled (LMAUTOEN=1) the PMC will switch automatically between Boot regulator and Last Mile regulator debending on the V15 status (LVD15S). 0b - Last Mile Regulator is off 1b - Last Mile Regulator is on
16 LMAUTOEN	Last Mile regulator auto turn over bit Enables to turn over automatically from Boot Regulator Mode to Last Mile regulator mode and vice versa depending on the V15 voltage status (LVD15S). As long as LMEN=0 software must make sure that the system clock is on FIRC clock or slower. To use higher clock speed software must set LMEN=1. 0b - Auto turnover disabled 1b - Auto turnover enabled
15-10 —	Reserved
9 LVDIE	Low voltage detect interrupt enable Enables hardware interrupt requests if any of the following flags is set: LVD5AF, LVD15F. LVD interrupt must be disabled before going into LPM. 0b - LVD hardware interrupt is disabled (use polling). 1b - Request an LVD hardware interrupt when LVDA5F=1 or LVD15F=1.
8 HVDIE	High voltage detect interrupt enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables hardware interrupt requests if any of the following flags is set: HVDAF, HVDBF, HVD25F, HVD11F. 0b - HVD hardware interrupt is disabled (use polling). 1b - Request an HVD hardware interrupt when HVDAF=1, HVDBF=1, HVD25F=1, or HVD11F=1.
7-5 —	Reserved
4 LVRBLPEN	LVRBLP enable bit during LPM Controls whether the low-voltage reset detection (LVRBLP) on the VDD_HV_B power domain is active or inactive in LPM 0b - Low-voltage reset detection is disabled in LPM. 1b - Low-voltage reset detection is enabled in LPM.
3 LPM25EN	V25 domain enable bit during LPM Controls whether the V25 regulator and low-voltage reset detection (LVR25LP) are active or inactive in LPM 0b - V25 regulator and LVR25LP are disabled in LPM. 1b - V25 regulator and LVR25LP are enabled in LPM.
2 FASTREC	Fast recovery from LPM enable bit Controls the recovery time from LPM to FPM. At recovery from LPM, all the tank capacitors from the secondary supplies have to be recharged. This causes a high-current demand, which might not be met by the supply driving the VDD_HV_A primary domain. When selecting the fast recovery time, the current for recharging is approximately three times higher than that for FASTREC=0. The application must determine from the drive capability of the external VDD_HV_A regulator, the size of tank caps on the secondary supply pins and the selected recovery time if this is sufficient to start up from LPM in time. 0b - Normal recovery time from LPM 1b - Fast recovery time from LPM
1 LMBCTLEN	Last Mile regulator base control enable bit This field must be set to 1 if external BJT between VDD_HV_A and V15 is used on the PCB. The base of this BJT must be connected to the VRC_CTRL pin and is controlled by the PMC to regulate a voltage of 1.5V on V15 pin. After setting LMBCTLEN=1 the software has to wait for 15us (FASTREC=1) respectively 50us (FASTREC=0) before polling LVD15S=0 and then setting LMEN=1. This respects the softstart time of the V15 regulator. If LMAUTOEN=1 then LMBCTLEN can be left enabled when going into LPM, the hardware will turn the regulator off and back on automatically after recovery from LPM. 0b - External BCTL regulator for V15 disabled 1b - External BCTL regulator for V15 enabled
0 LMEN	Last Mile regulator enable bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables the Last Mile regulator, which regulates an external 1.5V voltage on V15 down to the core and logic supply (V11 power domain), which is typically 1.1V. Setting LMEN=1 hands over the V11 voltage generation from the Boot regulator to the Last Mile regulator. The software must ensure that before enabling the Last Mile regulator, the voltage on V15 is sufficiently high as indicated by the LVD15S status field (LVD15S=0). To use external BJT between VDD_HV_A and V15, the LMBCTLEN field must be set before the LMEN field, and the software must wait until 1.5V is up (LVD15S=0). If LMAUTOEN=0 then to disable the Last Mile regulator, LMEN and LMBCTLEN must be cleared simultaneously (single register write). The software must disable (LMEN=0) the Last Mile regulator before going into LPM. After setting LMEN=1, software must wait a minimum time of 1.5us before changing clock rate. 0b - Last Mile regulator is disabled. 1b - Last Mile regulator is enabled.

41.7.4 Version ID register (VERID)

Offset

Register	Offset
VERID	Ch

Function

This register returns the major and minor version numbers of hardware implementation.

Diagram

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

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

Fields

Field	Function
31-24 MAJOR	Major version number Returns the version number for the specification

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-16 MINOR	Minor version number Returns the version number for the hardware implementation
15-1 —	Reserved
0 LMFEAT	Last Mile Regulator Feature This read-only field shows if the Last Mile regulator feature is available. 0b - No Last Mile regulator 1b - Last Mile regulator (1.5V to 1.1V) is available

41.8 Glossary

FPM	Full Performance mode
HVD	High voltage detect
IRC	Internal RC oscillator
LM	Last mile regulator
LPM	Low Performance mode
LVD	Low voltage detect
LVR	Low voltage reset
NVM	Nonvolatile memory
POR	Power on reset
XOSC	External crystal oscillator

Chapter 42
Power Management Controller (PMC for MWCT2016S)

42.1 Introduction

PMC provides multiple power options to allow you to optimize power consumption for the appropriate level of functionality. It includes:

- Internal voltage regulators
- POR
- Integrated low and high voltage detection system with reset (brownout) capability

The voltage regulator requires a 3.3 V or 5 V input to generate all the required secondary supplies.

42.2 Features

- A combination of internal and external voltage regulator options, offering Run and Standby modes
- Active POR, providing brownout detect
- LVR for all system-relevant power domains
- Software-readable Low Voltage Status and Control register that contains flags that indicate low- or high-voltage conditions

42.3 Modes of operation

PMC provides two basic modes of operation for voltage regulators and monitors:

- FPM supports chip run modes that have high current consumption.
- LPM supports chip standby modes that have low current consumption.

42.4 Block diagram

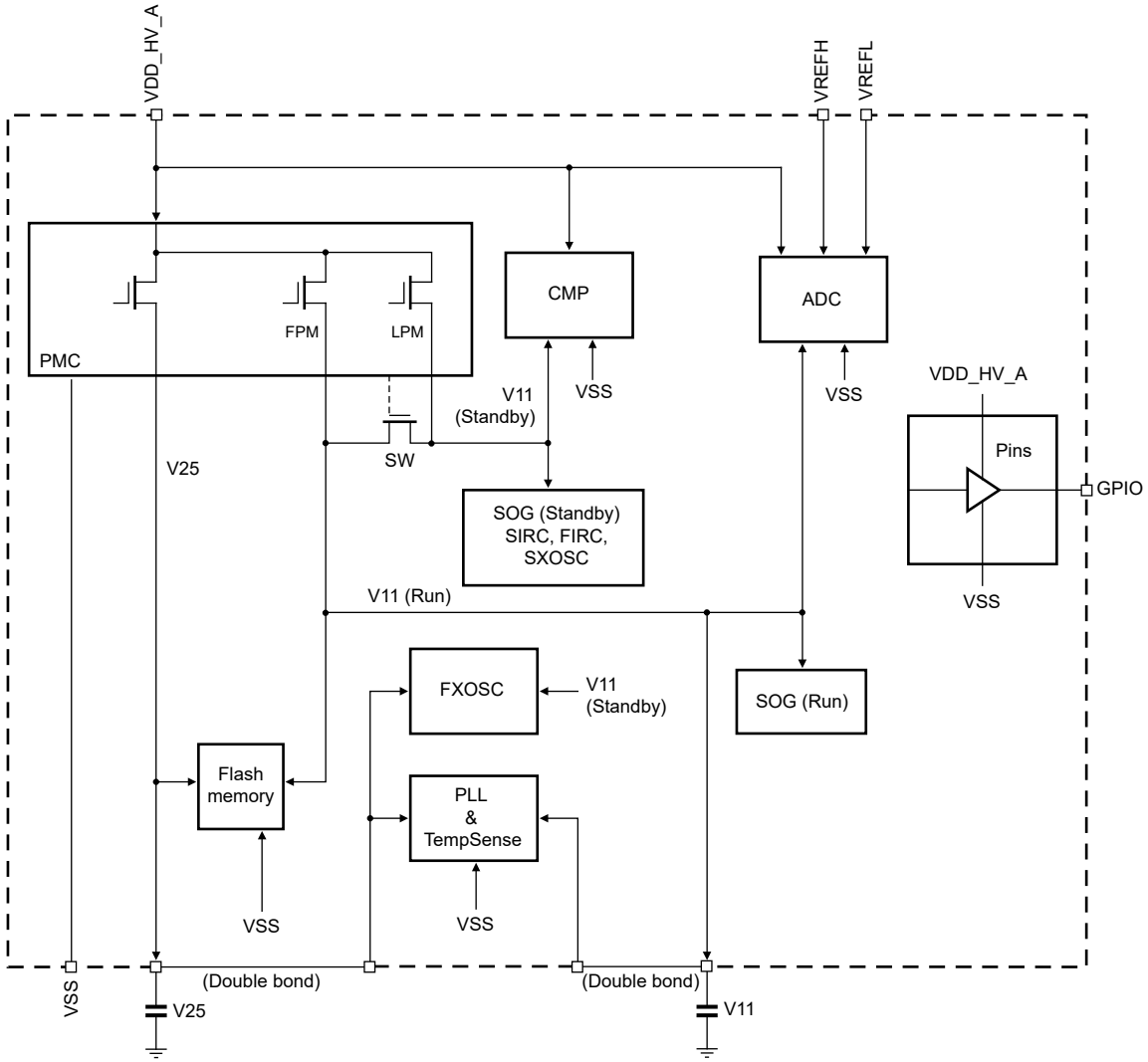


Figure 164. Block diagram

42.5 Signals

Table 214. Signals

Signal	Type	Description
VDD_HV_A	Supply input	The primary high-voltage supply input to PMC. VDD_HV_A is used for the PMC internal precision references. After the VDD_HV_A power domain is powered up, it must always be kept powered for both FPM and LPM.
V25	Supply output	A fully integrated low-dropout linear voltage regulator drives the V25 power supply domain. V25 supplies the flash memory and (via a double bond) the clock modules.

Table continues on the next page...

Table 214. Signals (continued)

Signal	Type	Description
V11	Supply output	V11 is the core and logic supply. A fully integrated low-dropout linear voltage regulator drives V11.
VSS	Ground	VSS must be grounded. You must connect all VSS pins externally to the same ground node.

42.6 Functional description

42.6.1 Reset

[BLG_PMC_RESET]The POR and all LVRs combine into one single chip POR.[end]

After a chip POR event, you can determine which power domain caused it by reading the PORF and LVR*x* flags in Low Voltage Status And Control (LVSC).

After an initial power ramp-up of the chip in LVSC, PMC sets the POR and LVR*x* flags. The go/no go flags have an arbitrary value.

NOTE

After an initial power ramp-up, you must clear all flags in LVSC by writing FFFF_FFFFh to that register.

Because the flags are sticky bits, you must clear them before using them. That way, if an unexpected chip POR occurs you can track and debug the source of the problem by reading the flags in LVSC.

42.6.2 Interrupts

PMC includes two interrupt sources:

- HVD interrupt: Combines all HVD monitors into one interrupt source. CONFIG[HVDIE] enables this interrupt. See PMC Configuration (CONFIG) and Low Voltage Status And Control (LVSC) for details.
- LVD interrupt: The LVD5A monitor is the only interrupt source. CONFIG[LVDIE] enables this interrupt. See PMC Configuration (CONFIG) and Low Voltage Status And Control (LVSC) for details.

42.7 PMC register descriptions

42.7.1 PMC memory map

PMC_MWCT2016S base address: 402E_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Low Voltage Status And Control (LVSC)	32	RW	See section
4h	PMC Configuration (CONFIG)	32	RW	0000_0000h
Ch	Version ID (VERID)	32	R	0300_0000h

42.7.2 Low Voltage Status And Control (LVSC)

Offset

Register	Offset
LVSC	0h

Function

Contains status and control fields that support the low-voltage reset and low- or high-voltage detect functions. When PMC is in LPM, the low- or high-voltage detect systems are disabled.

NOTE

For all flags that are not affected by reset (POR flag, all LVR flags, all GNG flags), if a reset occurs while trying to clear the flags (by writing 1), the flag value is not defined appropriately. In this case, you must clear the flag again after exiting from reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PORF	0					GNG11OS...	GNG25OS...	LVR11LPF	LVR11F	LVR25LPF	LVR25F	0		LVRALPF	LVRAF
W	W1C						W1C	W1C	W1C	W1C	W1C	W1C			W1C	W1C
Reset	u	0	0	0	0	0	u	u	u	u	u	u	0	0	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			LVD5AS	HVD11S	HVD25S	0	HVDAS	0			LVD5AF	HVD11F	HVD25F	0	HVDAF
W												W1C	W1C	W1C		W1C
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PORF	POR Flag [BLG_PMC_PORF] Indicates that a power-on reset event has occurred. Other reset sources have no effect. [end] **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
30-26 —	Reserved
25 GNG11OSCF	Go/No Go Detect Flag On OSC Part Of V11 Power Domain [BLG_PMC_GNG11OSCF] Indicates that the go/no go sensor has detected a low voltage in the V11 power domain in FPM. This applies only to the part of the power domain that supplies the 1.1 V clocking modules (for example, PLL). Other reset sources have no effect. [end] **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
24 GNG25OSCF	Go/No Go Detect Flag On OSC Part of V25 Power Domain [BLG_PMC_GNG25OSCF] Indicates that the go/no go sensor has detected a low voltage in the V25 power domain in FPM. This applies only to the part of the power domain that supplies the 2.5 V clocking modules (for example, XOSC and IRC). Other reset sources have no effect. [end] **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
23 LVR11LPF	LVR11LP Flag On V11 Power Domain [BLG_PMC_LVR11LPF] Indicates that a low-voltage reset event has occurred in the 1.1 V V11 power domain (FPM or LPM).Other reset sources have no effect. [end] **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
22 LVR11F	LVR11 Flag On V11 Power Domain In FPM [BLG_PMC_LVR11F] Indicates that a low-voltage reset event has occurred in the 1.1 V V11 power domain in FPM. Other reset sources have no effect. [end] **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
21 LVR25LPF	LVR25LP Flag On V25 Power Domain [BLG_PMC_LVR25LPF] Indicates that a low-voltage reset event has occurred in the V25 power domain (FPM or LPM). Other reset sources have no effect. [end] **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
20 LVR25F	LVR25 Flag On V25 Power Domain In FPM [BLG_PMC_LVR25F] Indicates that a low-voltage reset event has occurred in the V25 power domain in FPM. Other reset sources have no effect. [end] **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
19-18 —	Reserved
17 LVRALPF	LVRALP Flag On VDD_HV_A Power Domain [BLG_PMC_LVRALPF] Indicates that a low-voltage reset event has occurred in the VDD_HV_A power domain (FPM or LPM). Other reset sources have no effect. [end] **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
16 LVRAF	LVRA Flag On VDD_HV_A Power Domain In FPM [BLG_PMC_LVRAF] Indicates that a low-voltage reset event has occurred in the VDD_HV_A power domain in FPM. Other reset sources have no effect. [end] **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
15-13 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 LVD5AS	LVD5A Status On VDD_HV_A Power Domain In FPM [BLG_PMC_LVD5AS] Indicates whether the voltage on VDD_HV_A is above or below the low-voltage detect threshold. This monitor reflects the status of the 5 V low-voltage detect, LVD5A, and indicates if the voltage is below a certain threshold, which is set slightly below 4.5 V (see the chip data sheet for the exact value). The feature is only available in FPM and is disabled in LPM. After a reset or wake-up from LPM, you must clear the LVD5AF flag and check LVD5AS to determine the voltage level on the VDD_HV_A supply. [end] 0b - Above 1b - Below
11 HVD11S	HVD11 Status On V11 Power Domain In FPM [BLG_PMC_HVD11S] Indicates whether the voltage on V11 is above or below the high-voltage detect threshold. This field reflects the status of the high-voltage detect, HVD11, on the V11 power domain. This feature is only available in FPM and is disabled in LPM. [end] 0b - Voltage is below threshold or chip is in LPM 1b - Voltage is above threshold and chip is in FPM
10 HVD25S	HVD25 Status On V25 Power Domain In FPM [BLG_PMC_HVD25S] Indicates whether the voltage on V25 is above or below the high-voltage detect threshold. This field reflects the status of the high-voltage detect, HVD25, on the V25 power domain. The feature is only available in FPM and is disabled in LPM. [end] 0b - Voltage is below threshold or chip is in LPM 1b - Voltage is above threshold and chip is in FPM
9 —	Reserved
8 HVDAS	HVDA Status On VDD_HV_A Power Domain In FPM [BLG_PMC_HVDAS] Indicates whether the voltage on VDD_HV_A is above or below the high-voltage detect threshold. This field reflects the status of the high-voltage detect, HVDA, on the V25 power domain. The feature is only available in FPM and is disabled in LPM. [end] 0b - Voltage is below threshold or chip is in LPM 1b - Voltage is above threshold and chip is in FPM
7-5 —	Reserved
4 LVD5AF	LVD5A Flag On VDD_HV_A Power Domain In FPM [BLG_PMC_LVD5AF] Indicates whether LVD5AS has changed. When LVD5AS changes, PMC changes LVD5AF to 1. [end]

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Did not change 1b - Changed When writing 0b - No effect 1b - Clear the flag
3 HVD11F	HVD11 Flag On V11 Power Domain In FPM [BLG_PMC_HVD11F] Indicates whether HVD11S has changed. When HVD11S changes, PMC changes HVD11F to 1. [end] **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not change 1b - Changed When writing 0b - No effect 1b - Clear the flag
2 HVD25F	HVD25 Flag On V25 Power Domain In FPM [BLG_PMC_HVD25F] Indicates whether HVD25S has changed. When HVD25S changes, PMC changes HVD25F to 1. [end] **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not change 1b - Changed When writing 0b - No effect 1b - Clear the flag
1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 HVDAF	HVDA Flag On VDD_HV_A Power Domain In FPM [BLG_PMC_HVDAF] Indicates whether HVDAS has changed. When HVDAS changes, PMC changes HVDAF to 1. [end] **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not change 1b - Changed When writing 0b - No effect 1b - Clear the flag

42.7.3 PMC Configuration (CONFIG)

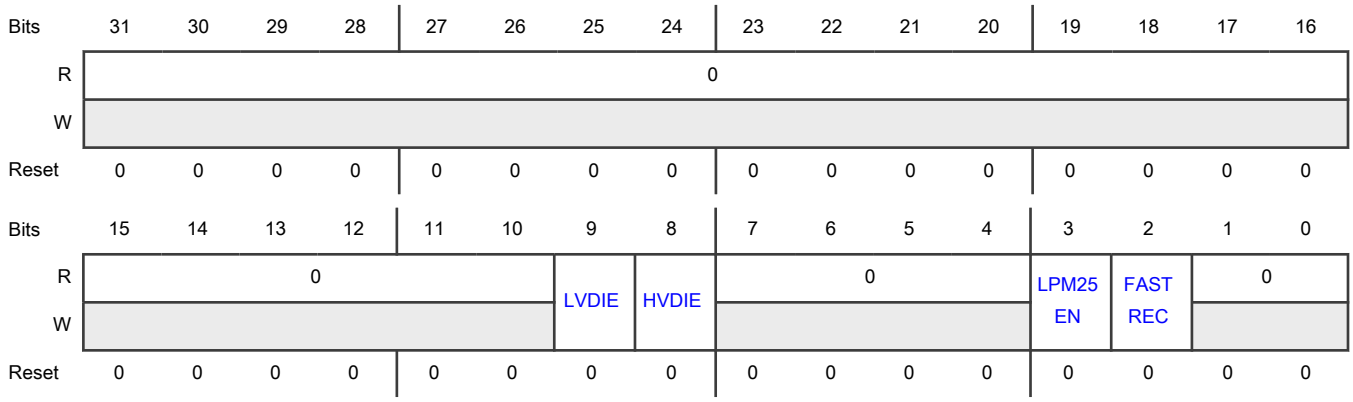
Offset

Register	Offset
CONFIG	4h

Function

Configures PMC.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 LVDIE	Low Voltage Detect Interrupt Enable [BLG_PMC_LVDIE] Enables hardware interrupt requests if LVSC[LVD5AF] = 1. You must disable the LVD interrupt before entering LPM. [end] 0b - LVD hardware interrupt is disabled (use polling) 1b - Request an LVD hardware interrupt when LVDA5F = 1
8 HVDIE	High Voltage Detect Interrupt Enable [BLG_PMC_HVDIE] Enables hardware interrupt requests if any of the following flags in Low Voltage Status And Control (LVSC) are set: • HVDAF • HVDBF • HVD25F • HVD11F [end] 0b - HVD hardware interrupt is disabled (use polling) 1b - Request an HVD hardware interrupt when HVDAF=1, HVDBF=1, HVD25F=1, or HVD11F=1
7-4 —	Reserved
3 LPM25EN	V25 Power Domain Enable During LPM [BLG_PMC_LPM25EN] Controls whether the V25 regulator and low-voltage reset detection (LVR25LP) are enabled or disabled in LPM [end] 0b - Disabled 1b - Enabled
2 FASTREC	Fast Recovery From LPM Enable [BLG_PMC_FASTREC] Controls the recovery time from LPM to FPM. At recovery from LPM, all the tank capacitors from the secondary supplies must be recharged. This causes a high-current demand, which the supply driving the VDD_HV_A primary power domain might not meet. When you select the fast recovery time, the current for recharging is approximately three times higher than that for normal recovery time. You must determine whether this current is sufficient to start up from LPM in time, using these criteria: • Drive capability of the external VDD_HV_A regulator • Size of the tank caps on the secondary supply pin • Selected recovery time

Table continues on the next page...

Table continued from the previous page...

Field	Function
	[end] 0b - Normal 1b - Fast
1-0 —	Reserved

42.7.4 Version ID (VERID)

Offset

Register	Offset
VERID	Ch

Function

Records the specific PMC version in the chip.

Diagram

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

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

Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number of the PMC design.
23-16 MINOR	Minor Version Number Indicates the minor version number of the PMC design.
15-1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 LMFEAT	Last-Mile Regulator Feature Indicates whether the last-mile regulator (1.5 V to 1.1 V) is available. 0b - Not available 1b - Available

42.8 Glossary

FPM	Full Performance mode
HVD	High voltage detect
LPM	Low Power mode
LVD	Low voltage detect
LVR	Low voltage reset
POR	Power-on reset

Chapter 43
Mode Entry Module (MC_ME)

43.1 Chip-specific MC_ME information

43.1.1 MC_ME modes

This chip implements these modes:

- Reset
- Run
- Standby

The chip always enters Run mode after exiting Reset mode wherein you can configure the chip to perform its computational and communication functions. In Run mode:

- The chip remains fully powered. The boot core is the only core enabled on Run entry.
- You can enable and disable application cores and peripherals as needed, based on functional and power requirements.
- You can configure the pins and self-test as needed. See "Self-Test programming sequence" section in the STCU2 chapter for the self-test programming sequence to be followed before initiating self-test, in which:
  - Pins are safe-stated.
  - No computational or communication activities are possible.
- The chip automatically enters Reset mode after self-test (BIST) is complete.

43.1.2 Core operation modes

The Cortex-M7 cores in the chip support these two modes of operation:

- Decoupled/independent operation
- Coupled/lockstep operation

The device configuration clients (utest_misc DCF client) control these modes of operation. See the DCF clients file attached to this document for details.

43.1.3 MC_ME partition mapping of cores and peripherals

MC_ME provides registers and interface signals to support multiple partitions. These MC_ME partitions are different from the chip's LBIST partitions described in the "Safety Overview" chapter. This chip has three MC_ME partitions:

- Partition 0: Contains application cores and the on/off-platform slots on AIPS_0 bridge
- Partition 1: Contains the on/off-platform slots on AIPS_1 bridge
- Partition 2: Contains the on/off-platform slots on AIPS_2 bridge

Table 215 and Table 216 specify the core and peripheral mapping on MC_ME partitions and their associated clock gating possiblities.

MC_ME also provides provisions to control the booting address for application cores, which can be configured to start from a nondefault address location by appropriately configuring PRTN*x*_CORE*n*_ADDR[ADDR].

43.1.4 Core clock gating

MC_ME has individual core-clock enable fields that gate the application core clocks, which can also be clock gated by executing Waiting for Interrupt (WFI).

You can enable the application cores by configuring the respective CCE fields. See Application core initialization process and Application core shutdown process for proper initialization and shutdown of application cores. There is no clock control for the HSE_B core. It needs to be only put into WFI if required to be shutdown (for example, in Standby mode).

Table 215. MC_ME partition core mapping

Core	MC_ME partition	MC_ME clock enable register field
Cortex-M7_0	0	MC_ME.PRTN0_CORE0_PCONF[CCE]
Cortex-M7_1 [1]	0	MC_ME.PRTN0_CORE1_PCONF[CCE]

1. Not present on MWCT2016S and MWCT2015S

43.1.5 Peripheral clock gating

See Peripheral initialization process and Peripheral shutdown process for proper initialization and shutdown of peripherals. See Table 216 for peripheral clock gating possibilities.

The application core can program the reserved configurations.

NOTE

Before accessing the registers of a peripheral to start using it, its clock must be turned on, otherwise, a Hard-Fault event will occur.

Table 216. MC_ME partition peripheral mapping and clock control

AIPS peripheral	Peripheral description[1]	MC_ME's PRTN*n*_COFB*n*_CLKEN register fields	MC_ME peripheral control register	MC_ME peripheral slot number in partition	MC_ME COFB present	MC_ME CLKEN present	MC_ME default CLKIN	On platform	IPS slot number
0	Reserved	PRTN0_COFB0_CLKEN[REQ0]	0	0	0	0	0	Yes	0
0	Reserved	PRTN0_COFB0_CLKEN[REQ1]	1	1	0	0	0	Yes	1
0	Reserved	PRTN0_COFB0_CLKEN[REQ2]	2	2	0	0	0	Yes	2
0	Reserved	PRTN0_COFB0_CLKEN[REQ3]	3	3	0	0	0	Yes	3
0	Reserved	PRTN0_COFB0_CLKEN[REQ4]	4	4	0	0	0	Yes	4
0	Reserved	PRTN0_COFB0_CLKEN[REQ5]	5	5	0	0	0	Yes	5
0	Reserved	PRTN0_COFB0_CLKEN[REQ6]	6	6	0	0	0	Yes	6
0	Reserved	PRTN0_COFB0_CLKEN[REQ7]	7	7	0	0	0	Yes	7
0	Reserved	PRTN0_COFB0_CLKEN[REQ8]	8	8	0	0	0	Yes	8
0	Reserved	PRTN0_COFB0_CLKEN[REQ9]	9	9	0	0	0	Yes	9
0	Reserved	PRTN0_COFB0_CLKEN[REQ10]	10	10	0	0	0	Yes	10
0	Reserved	PRTN0_COFB0_CLKEN[REQ11]	11	11	0	0	0	Yes	11
0	Reserved	PRTN0_COFB0_CLKEN[REQ12]	12	12	0	0	0	Yes	12
0	Reserved	PRTN0_COFB0_CLKEN[REQ13]	13	13	0	0	0	Yes	13
0	Reserved	PRTN0_COFB0_CLKEN[REQ14]	14	14	0	0	0	Yes	14
0	Reserved	PRTN0_COFB0_CLKEN[REQ15]	15	15	0	0	0	Yes	15
0	Reserved	PRTN0_COFB0_CLKEN[REQ16]	16	16	0	0	0	Yes	16
0	Reserved	PRTN0_COFB0_CLKEN[REQ17]	17	17	0	0	0	Yes	17
0	Reserved	PRTN0_COFB0_CLKEN[REQ18]	18	18	0	0	0	Yes	18
0	Reserved	PRTN0_COFB0_CLKEN[REQ19]	19	19	0	0	0	Yes	19
0	Reserved	PRTN0_COFB0_CLKEN[REQ20]	20	20	0	0	0	Yes	20

Table continues on the next page...

Table 216. MC_ME partition peripheral mapping and clock control (continued)

AIPS peripheral	Peripheral description[1]	MC_ME's PRTN*n*_COFB*n*_CLKEN register fields	MC_ME peripheral control register	MC_ME peripheral slot number in partition	MC_ME COFB present	MC_ME CLKEN present	MC_ME default CLKIN	On platform	IPS slot number
0	Reserved	PRTN0_COFB0_CLKEN[REQ21]	21	21	0	0	0	Yes	21
0	Reserved	PRTN0_COFB0_CLKEN[REQ22]	22	22	0	0	0	Yes	22
0	Reserved	PRTN0_COFB0_CLKEN[REQ23]	23	23	0	0	0	Yes	23
0	Reserved	PRTN0_COFB0_CLKEN[REQ24]	24	24	0	0	0	Yes	24
0	Reserved	PRTN0_COFB0_CLKEN[REQ25]	25	25	0	0	0	Yes	25
0	Reserved	PRTN0_COFB0_CLKEN[REQ26]	26	26	0	0	0	Yes	26
0	Reserved	PRTN0_COFB0_CLKEN[REQ27]	27	27	0	0	0	Yes	27
0	Reserved	PRTN0_COFB0_CLKEN[REQ28]	28	28	0	0	0	Yes	28
0	Reserved	PRTN0_COFB0_CLKEN[REQ29]	29	29	0	0	0	Yes	29
0	Reserved	PRTN0_COFB0_CLKEN[REQ30]	30	30	0	0	0	Yes	30
0	Reserved	PRTN0_COFB0_CLKEN[REQ31]	31	31	0	0	0	Yes	31
0	TRGMUX	PRTN0_COFB1_CLKEN[REQ32]	32	32	1	1	0	No	0
0	BCTU	PRTN0_COFB1_CLKEN[REQ33]	33	33	1	1	0	No	1
0	eMIOS_0	PRTN0_COFB1_CLKEN[REQ34]	34	34	1	1	0	No	2
0	eMIOS_1	PRTN0_COFB1_CLKEN[REQ35]	35	35	1	1	0	No	3
0	eMIOS_2	PRTN0_COFB1_CLKEN[REQ36]	36	36	1	1	0	No	4
0	Reserved	PRTN0_COFB1_CLKEN[REQ37]	37	37	0	0	0	No	5
0	LCU_0	PRTN0_COFB1_CLKEN[REQ38]	38	38	1	1	0	No	6
0	LCU_1	PRTN0_COFB1_CLKEN[REQ39]	39	39	1	1	0	No	7
0	ADC_0	PRTN0_COFB1_CLKEN[REQ40]	40	40	1	1	0	No	8
0	ADC_1	PRTN0_COFB1_CLKEN[REQ41]	41	41	1	1	0	No	9

Table continues on the next page...

Table 216. MC_ME partition peripheral mapping and clock control (continued)

AIPS peripheral	Peripheral description[1]	MC_ME's PRTN*n*_COFB*n*_CLKEN register fields	MC_ME peripheral control register	MC_ME peripheral slot number in partition	MC_ME COFB present	MC_ME CLKEN present	MC_ME default CLKIN	On platform	IPS slot number
0	ADC_2	PRTN0_COFB1_CLKEN[REQ42]	42	42	1	1	0	No	10
0	Reserved	PRTN0_COFB1_CLKEN[REQ43]	43	43	0	0	0	No	11
0	PIT_0	PRTN0_COFB1_CLKEN[REQ44]	44	44	1	1	1	No	12
0	PIT_1	PRTN0_COFB1_CLKEN[REQ45]	45	45	1	1	0	No	13
0	MU_A	PRTN0_COFB1_CLKEN[REQ46]	46	46	1	1	0	No	14
0	MU_B	PRTN0_COFB1_CLKEN[REQ47]	47	47	1	1	0	No	15
1	AXBS switch	PRTN1_COFB0_CLKEN[REQ0]	128	0	1	0	1	Yes	0
1	AXBS_0	PRTN1_COFB0_CLKEN[REQ1]	129	1	1	0	1	Yes	1
1	AXBS_1	PRTN1_COFB0_CLKEN[REQ2]	130	2	1	0	1	Yes	2
1	eDMA control and status (MP_CSR; MP_ES; MP_HRS)	PRTN1_COFB0_CLKEN[REQ3]	131	3	1	1	0	Yes	3
1	eDMA transfer control descriptor 0	PRTN1_COFB0_CLKEN[REQ4]	132	4	1	1	0	Yes	4
1	eDMA transfer control descriptor 1	PRTN1_COFB0_CLKEN[REQ5]	133	5	1	1	0	Yes	5
1	eDMA transfer control descriptor 2	PRTN1_COFB0_CLKEN[REQ6]	134	6	1	1	0	Yes	6
1	eDMA transfer control descriptor 3	PRTN1_COFB0_CLKEN[REQ7]	135	7	1	1	0	Yes	7
1	eDMA transfer control descriptor 4	PRTN1_COFB0_CLKEN[REQ8]	136	8	1	1	0	Yes	8
1	eDMA transfer control descriptor 5	PRTN1_COFB0_CLKEN[REQ9]	137	9	1	1	0	Yes	9

Table continues on the next page...

Table 216. MC_ME partition peripheral mapping and clock control (continued)

AIPS peripheral	Peripheral description[1]	MC_ME's PRTN*n*_COFB*n*_CLKEN register fields	MC_ME peripheral control register	MC_ME peripheral slot number in partition	MC_ME COFB present	MC_ME CLKEN present	MC_ME default CLKIN	On platform	IPS slot number
1	eDMA transfer control descriptor 6	PRTN1_COFB0_CLKEN[REQ10]	138	10	1	1	0	Yes	10
1	eDMA transfer control descriptor 7	PRTN1_COFB0_CLKEN[REQ11]	139	11	1	1	0	Yes	11
1	eDMA transfer control descriptor 8	PRTN1_COFB0_CLKEN[REQ12]	140	12	1	1	0	Yes	12
1	eDMA transfer control descriptor 9	PRTN1_COFB0_CLKEN[REQ13]	141	13	1	1	0	Yes	13
1	eDMA transfer control descriptor 10	PRTN1_COFB0_CLKEN[REQ14]	142	14	1	1	0	Yes	14
1	eDMA transfer control descriptor 11	PRTN1_COFB0_CLKEN[REQ15]	143	15	1	1	0	Yes	15
1	Debug APB page0	PRTN1_COFB0_CLKEN[REQ16]	144	16	1	0	1	Yes	16
1	Debug APB page1	PRTN1_COFB0_CLKEN[REQ17]	145	17	1	0	1	Yes	17
1	Debug APB page2	PRTN1_COFB0_CLKEN[REQ18]	146	18	1	0	1	Yes	18
1	Debug APB page3	PRTN1_COFB0_CLKEN[REQ19]	147	19	1	0	1	Yes	19
1	Debug APB paged area	PRTN1_COFB0_CLKEN[REQ20]	148	20	1	0	1	Yes	20
1	SDA_AP	PRTN1_COFB0_CLKEN[REQ21]	149	21	1	1	1	Yes	21
1	EIM	PRTN1_COFB0_CLKEN[REQ22]	150	22	1	1	0	Yes	22
1	ERM	PRTN1_COFB0_CLKEN[REQ23]	151	23	1	1	0	Yes	23
1	MSCM	PRTN1_COFB0_CLKEN[REQ24]	152	24	1	1	0	Yes	24
1	PRAMC_0	PRTN1_COFB0_CLKEN[REQ25]	153	25	1	0	1	Yes	25
1	Flash memory controller	PRTN1_COFB0_CLKEN[REQ26]	154	26	1	0	1	Yes	26

Table continues on the next page...

Table 216. MC_ME partition peripheral mapping and clock control (continued)

AIPS peripheral	Peripheral description[1]	MC_ME's PRTN*n*_COFB*n*_CLKEN register fields	MC_ME peripheral control register	MC_ME peripheral slot number in partition	MC_ME COFB present	MC_ME CLKEN present	MC_ME default CLKIN	On platform	IPS slot number
1	Flash memory controller alternate	PRTN1_COFB0_CLKEN[REQ27]	155	27	1	0	1	Yes	27
1	SWT_0	PRTN1_COFB0_CLKEN[REQ28]	156	28	1	1	1	Yes	28
1	STM_0	PRTN1_COFB0_CLKEN[REQ29]	157	29	1	1	0	Yes	29
1	XRDC	PRTN1_COFB0_CLKEN[REQ30]	158	30	1	0	1	Yes	30
1	INTM	PRTN1_COFB0_CLKEN[REQ31]	159	31	1	1	0	Yes	31
1	DMAMUX_0	PRTN1_COFB1_CLKEN[REQ32]	160	32	1	1	0	No	0
1	DMAMUX_1	PRTN1_COFB1_CLKEN[REQ33]	161	33	1	1	0	No	1
1	RTC	PRTN1_COFB1_CLKEN[REQ34]	162	34	1	1	1	No	2
1	MC_RGM	PRTN1_COFB1_CLKEN[REQ35]	163	35	1	0	1	No	3
1	SIUL2_VIRTWRAPPER_PDAC0	PRTN1_COFB1_CLKEN[REQ36]	164	36	1	0	1	No	4
1	SIUL2_VIRTWRAPPER_PDAC0	PRTN1_COFB1_CLKEN[REQ37]	165	37	1	0	1	No	5
1	SIUL2_VIRTWRAPPER_PDAC1	PRTN1_COFB1_CLKEN[REQ38]	166	38	1	0	1	No	6
1	SIUL2_VIRTWRAPPER_PDAC1	PRTN1_COFB1_CLKEN[REQ39]	167	39	1	0	1	No	7
1	SIUL2_VIRTWRAPPER_PDAC2	PRTN1_COFB1_CLKEN[REQ40]	168	40	1	0	1	No	8
1	SIUL2_VIRTWRAPPER_PDAC2	PRTN1_COFB1_CLKEN[REQ41]	169	41	1	0	1	No	9
1	SIUL2_VIRTWRAPPER	PRTN1_COFB1_CLKEN[REQ42]	170	42	1	1	1	No	10
1	SSCM	PRTN1_COFB1_CLKEN[REQ43]	171	43	1	0	1	No	11

Table continues on the next page...

Table 216. MC_ME partition peripheral mapping and clock control (continued)

AIPS peripheral	Peripheral description[1]	MC_ME's PRTN*n*_COFB*n*_CLKEN register fields	MC_ME peripheral control register	MC_ME peripheral slot number in partition	MC_ME COFB present	MC_ME CLKEN present	MC_ME default CLKIN	On platform	IPS slot number
1	Reserved	PRTN1_COFB1_CLKEN[REQ44]	172	44	0	0	0	No	12
1	WKPU	PRTN1_COFB1_CLKEN[REQ45]	173	45	1	1	1	No	13
1	Reserved	PRTN1_COFB1_CLKEN[REQ46]	174	46	0	0	0	No	14
1	CMU 0-5	PRTN1_COFB1_CLKEN[REQ47]	175	47	1	1	0	No	15
1	Reserved	PRTN1_COFB1_CLKEN[REQ48]	176	48	0	0	0	No	16
1	TSPC	PRTN1_COFB1_CLKEN[REQ49]	177	49	1	1	1	No	17
1	32 kHz SIRC	PRTN1_COFB1_CLKEN[REQ50]	178	50	1	0	1	No	18
1	32 kHz SXOSC	PRTN1_COFB1_CLKEN[REQ51]	179	51	1	1	1	No	19
1	48 MHz FIRC	PRTN1_COFB1_CLKEN[REQ52]	180	52	1	0	1	No	20
1	8-40 MHz FXOSC	PRTN1_COFB1_CLKEN[REQ53]	181	53	1	1	1	No	21
1	MC_CGM	PRTN1_COFB1_CLKEN[REQ54]	182	54	1	0	1	No	22
1	MC_ME	PRTN1_COFB1_CLKEN[REQ55]	183	55	1	0	1	No	23
1	Frequency-modulated PLL	PRTN1_COFB1_CLKEN[REQ56]	184	56	1	1	0	No	24
1	Reserved	PRTN1_COFB1_CLKEN[REQ57]	185	57	0	0	0	No	25
1	PMC	PRTN1_COFB1_CLKEN[REQ58]	186	58	1	0	1	No	26
1	Flash memory controller	PRTN1_COFB1_CLKEN[REQ59]	187	59	1	0	1	No	27
1	Flash memory controller alternate	PRTN1_COFB1_CLKEN[REQ60]	188	60	1	0	1	No	28
1	Reserved	PRTN1_COFB1_CLKEN[REQ61]	189	61	0	0	0	No	29
1	Reserved	PRTN1_COFB1_CLKEN[REQ62]	190	62	0	0	0	No	30
1	PIT_2	PRTN1_COFB1_CLKEN[REQ63]	191	63	1	1	0	No	31

Table continues on the next page...

Table 216. MC_ME partition peripheral mapping and clock control (continued)

AIPS peripheral	Peripheral description[1]	MC_ME's PRTN*n*_COFB*n*_CLKEN register fields	MC_ME peripheral control register	MC_ME peripheral slot number in partition	MC_ME COFB present	MC_ME CLKEN present	MC_ME default CLKIN	On platform	IPS slot number
1	Reserved	PRTN1_COFB2_CLKEN[REQ64]	192	64	0	0	0	No	32
1	FlexCAN_0	PRTN1_COFB2_CLKEN[REQ65]	193	65	1	1	0	No	33
1	FlexCAN_1	PRTN1_COFB2_CLKEN[REQ66]	194	66	1	1	0	No	34
1	FlexCAN_2	PRTN1_COFB2_CLKEN[REQ67]	195	67	1	1	0	No	35
1	FlexCAN_3	PRTN1_COFB2_CLKEN[REQ68]	196	68	1	1	0	No	36
1	FlexCAN_4	PRTN1_COFB2_CLKEN[REQ69]	197	69	1	1	0	No	37
1	FlexCAN_5	PRTN1_COFB2_CLKEN[REQ70]	198	70	1	1	0	No	38
1	Reserved	PRTN1_COFB2_CLKEN[REQ71]	199	71	0	0	0	No	39
1	Reserved	PRTN1_COFB2_CLKEN[REQ72]	200	72	0	0	0	No	40
1	FlexIO	PRTN1_COFB2_CLKEN[REQ73]	201	73	1	1	0	No	41
1	LPUART_0	PRTN1_COFB2_CLKEN[REQ74]	202	74	1	1	0	No	42
1	LPUART_1	PRTN1_COFB2_CLKEN[REQ75]	203	75	1	1	0	No	43
1	LPUART_2	PRTN1_COFB2_CLKEN[REQ76]	204	76	1	1	0	No	44
1	LPUART_3	PRTN1_COFB2_CLKEN[REQ77]	205	77	1	1	0	No	45
1	LPUART_4	PRTN1_COFB2_CLKEN[REQ78]	206	78	1	1	0	No	46
1	LPUART_5	PRTN1_COFB2_CLKEN[REQ79]	207	79	1	1	0	No	47
1	LPUART_6	PRTN1_COFB2_CLKEN[REQ80]	208	80	1	1	0	No	48
1	LPUART_7	PRTN1_COFB2_CLKEN[REQ81]	209	81	1	1	0	No	49
1	Reserved	PRTN1_COFB2_CLKEN[REQ82]	210	82	0	0	0	No	50
1	Reserved	PRTN1_COFB2_CLKEN[REQ83]	211	83	0	0	0	No	51
1	LPI2C_0	PRTN1_COFB2_CLKEN[REQ84]	212	84	1	1	0	No	52

Table continues on the next page...

Table 216. MC_ME partition peripheral mapping and clock control (continued)

AIPS peripheral	Peripheral description[1]	MC_ME's PRTN*n*_COFB*n*_CLKEN register fields	MC_ME peripheral control register	MC_ME peripheral slot number in partition	MC_ME COFB present	MC_ME CLKEN present	MC_ME default CLKIN	On platform	IPS slot number
1	LPI2C_1	PRTN1_COFB2_CLKEN[REQ85]	213	85	1	1	0	No	53
1	LPSPI_0	PRTN1_COFB2_CLKEN[REQ86]	214	86	1	1	0	No	54
1	LPSPI_1	PRTN1_COFB2_CLKEN[REQ87]	215	87	1	1	0	No	55
1	LPSPI_2	PRTN1_COFB2_CLKEN[REQ88]	216	88	1	1	0	No	56
1	LPSPI_3	PRTN1_COFB2_CLKEN[REQ89]	217	89	1	1	0	No	57
1	Reserved	PRTN1_COFB2_CLKEN[REQ90]	218	90	0	0	0	No	58
1	SAI_0	PRTN1_COFB2_CLKEN[REQ91]	219	91	1	1	0	No	59
1	LPCMP_0	PRTN1_COFB2_CLKEN[REQ92]	220	92	1	1	1	No	60
1	LPCMP_1	PRTN1_COFB2_CLKEN[REQ93]	221	93	1	1	1	No	61
1	Reserved	PRTN1_COFB2_CLKEN[REQ94]	222	94	0	0	0	No	62
1	TempSense	PRTN1_COFB2_CLKEN[REQ95]	223	95	1	1	0	No	63
1	CRC	PRTN1_COFB3_CLKEN[REQ96]	224	96	1	1	0	No	64
1	FCCU (+FOSU)	PRTN1_COFB3_CLKEN[REQ97]	225	97	1	0	1	No	65
1	MTR	PRTN1_COFB3_CLKEN[REQ98]	226	98	1	0	1	No	66
1	HSE_MU0_B	PRTN1_COFB3_CLKEN[REQ99]	227	99	1	0	1	No	67
1	Reserved	PRTN1_COFB3_CLKEN[REQ100]	228	100	0	0	0	No	68
1	JDC	PRTN1_COFB3_CLKEN[REQ101]	229	101	1	0	1	No	69
1	Reserved	PRTN1_COFB3_CLKEN[REQ102]	230	102	1	1	1	No	70
1	Configuration GPR	PRTN1_COFB3_CLKEN[REQ103]	231	103	1	0	1	No	71
1	STCU2	PRTN1_COFB3_CLKEN[REQ104]	232	104	1	1	1	No	72
1	Reserved	PRTN1_COFB3_CLKEN[REQ105]	233	105	1	0	1	No	73

Table continues on the next page...

Table 216. MC_ME partition peripheral mapping and clock control (continued)

AIPS peripheral	Peripheral description[1]	MC_ME's PRTN*n*_COFB*n*_CLKEN register fields	MC_ME peripheral control register	MC_ME peripheral slot number in partition	MC_ME COFB present	MC_ME CLKEN present	MC_ME default CLKIN	On platform	IPS slot number
1	Reserved	PRTN1_COFB3_CLKEN[REQ106]	234	106	1	0	1	No	74
1	Reserved	PRTN1_COFB3_CLKEN[REQ107]	235	107	1	0	1	No	75
1	SELFTEST_GPR[2]	PRTN1_COFB3_CLKEN[REQ108]	236	108	1	0	1	No	76
1	Reserved	PRTN1_COFB3_CLKEN[REQ109]	237	109	0	0	0	No	77
1	Reserved	PRTN1_COFB3_CLKEN[REQ110]	238	110	1	0	1	No	78
2	XBIC (TCM backdoor AHB_splitter)	PRTN2_COFB0_CLKEN[REQ0]	256	0	1	0	1	Yes	0
2	XBIC (eDMA AXBS-Lite)	PRTN2_COFB0_CLKEN[REQ1]	257	1	1	0	1	Yes	1
2	Reserved	PRTN2_COFB0_CLKEN[REQ2]	258	2	0	0	0	Yes	2
2	Reserved	PRTN2_COFB0_CLKEN[REQ3]	259	3	0	0	0	Yes	3
2	eDMA transfer control descriptor 12	PRTN2_COFB0_CLKEN[REQ4]	260	4	1	1	0	Yes	4
2	eDMA transfer control descriptor 13	PRTN2_COFB0_CLKEN[REQ5]	261	5	1	1	0	Yes	5
2	eDMA transfer control descriptor 14	PRTN2_COFB0_CLKEN[REQ6]	262	6	1	1	0	Yes	6
2	eDMA transfer control descriptor 15	PRTN2_COFB0_CLKEN[REQ7]	263	7	1	1	0	Yes	7
2	eDMA transfer control descriptor 16	PRTN2_COFB0_CLKEN[REQ8]	264	8	1	1	0	Yes	8
2	eDMA transfer control descriptor 17	PRTN2_COFB0_CLKEN[REQ9]	265	9	1	1	0	Yes	9
2	eDMA transfer control descriptor 18	PRTN2_COFB0_CLKEN[REQ10]	266	10	1	1	0	Yes	10

Table continues on the next page...

Table 216. MC_ME partition peripheral mapping and clock control (continued)

AIPS peripheral	Peripheral description[1]	MC_ME's PRTN*n*_COFB*n*_CLKEN register fields	MC_ME peripheral control register	MC_ME peripheral slot number in partition	MC_ME COFB present	MC_ME CLKEN present	MC_ME default CLKIN	On platform	IPS slot number
2	eDMA transfer control descriptor 19	PRTN2_COFB0_CLKEN[REQ11]	267	11	1	1	0	Yes	11
2	eDMA transfer control descriptor 20	PRTN2_COFB0_CLKEN[REQ12]	268	12	1	1	0	Yes	12
2	eDMA transfer control descriptor 21	PRTN2_COFB0_CLKEN[REQ13]	269	13	1	1	0	Yes	13
2	eDMA transfer control descriptor 22	PRTN2_COFB0_CLKEN[REQ14]	270	14	1	1	0	Yes	14
2	eDMA transfer control descriptor 23	PRTN2_COFB0_CLKEN[REQ15]	271	15	1	1	0	Yes	15
2	eDMA transfer control descriptor 24	PRTN2_COFB0_CLKEN[REQ16]	272	16	1	1	0	Yes	16
2	eDMA transfer control descriptor 25	PRTN2_COFB0_CLKEN[REQ17]	273	17	1	1	0	Yes	17
2	eDMA transfer control descriptor 26	PRTN2_COFB0_CLKEN[REQ18]	274	18	1	1	0	Yes	18
2	eDMA transfer control descriptor 27	PRTN2_COFB0_CLKEN[REQ19]	275	19	1	1	0	Yes	19
2	eDMA transfer control descriptor 28	PRTN2_COFB0_CLKEN[REQ20]	276	20	1	1	0	Yes	20
2	eDMA transfer control descriptor 29	PRTN2_COFB0_CLKEN[REQ21]	277	21	1	1	0	Yes	21
2	eDMA transfer control descriptor 30	PRTN2_COFB0_CLKEN[REQ22]	278	22	1	1	0	Yes	22
2	eDMA transfer control descriptor 31	PRTN2_COFB0_CLKEN[REQ23]	279	23	1	1	0	Yes	23

Table continues on the next page...

Table 216. MC_ME partition peripheral mapping and clock control (continued)

AIPS peripheral	Peripheral description[1]	MC_ME's PRTN*n*_COFB*n*_CLKEN register fields	MC_ME peripheral control register	MC_ME peripheral slot number in partition	MC_ME COFB present	MC_ME CLKEN present	MC_ME default CLKIN	On platform	IPS slot number
2	SEMA42	PRTN2_COFB0_CLKEN[REQ24]	280	24	1	1	0	Yes	24
2	PRAMC_1	PRTN2_COFB0_CLKEN[REQ25]	281	25	1	0	1	Yes	25
2	Reserved	PRTN2_COFB0_CLKEN[REQ26]	282	26	0	0	0	Yes	26
2	SWT_1	PRTN2_COFB0_CLKEN[REQ27]	283	27	1	1	0	Yes	27
2	Reserved	PRTN2_COFB0_CLKEN[REQ28]	284	28	0	0	0	Yes	28
2	STM_1	PRTN2_COFB0_CLKEN[REQ29]	285	29	1	1	0	Yes	29
2	Reserved	PRTN2_COFB0_CLKEN[REQ30]	286	30	0	0	0	Yes	30
2	Reserved	PRTN2_COFB0_CLKEN[REQ31]	287	31	0	0	0	Yes	31
2	EMAC	PRTN2_COFB1_CLKEN[REQ32]	288	32	1	1	0	No	0
2	Reserved	PRTN2_COFB1_CLKEN[REQ33]	289	33	0	0	0	No	1
2	Reserved	PRTN2_COFB1_CLKEN[REQ34]	290	34	0	0	0	No	2
2	LPUART_8	PRTN2_COFB1_CLKEN[REQ35]	291	35	1	1	0	No	3
2	LPUART_9	PRTN2_COFB1_CLKEN[REQ36]	292	36	1	1	0	No	4
2	LPUART_10	PRTN2_COFB1_CLKEN[REQ37]	293	37	1	1	0	No	5
2	LPUART_11	PRTN2_COFB1_CLKEN[REQ38]	294	38	1	1	0	No	6
2	LPUART_12	PRTN2_COFB1_CLKEN[REQ39]	295	39	1	1	0	No	7
2	LPUART_13	PRTN2_COFB1_CLKEN[REQ40]	296	40	1	1	0	No	8
2	LPUART_14	PRTN2_COFB1_CLKEN[REQ41]	297	41	1	1	0	No	9
2	LPUART_15	PRTN2_COFB1_CLKEN[REQ42]	298	42	1	1	0	No	10
2	Reserved	PRTN2_COFB1_CLKEN[REQ43]	299	43	0	0	0	No	11
2	Reserved	PRTN2_COFB1_CLKEN[REQ44]	300	44	0	0	0	No	12

Table continues on the next page...

Table 216. MC_ME partition peripheral mapping and clock control (continued)

AIPS peripheral	Peripheral description[1]	MC_ME's PRTN*n*_COFB*n*_CLKEN register fields	MC_ME peripheral control register	MC_ME peripheral slot number in partition	MC_ME COFB present	MC_ME CLKEN present	MC_ME default CLKIN	On platform	IPS slot number
2	Reserved	PRTN2_COFB1_CLKEN[REQ45]	301	45	0	0	0	No	13
2	Reserved	PRTN2_COFB1_CLKEN[REQ46]	302	46	0	0	0	No	14
2	LPSPI_4	PRTN2_COFB1_CLKEN[REQ47]	303	47	1	1	0	No	15
2	LPSPI_5	PRTN2_COFB1_CLKEN[REQ48]	304	48	1	1	0	No	16
2	Reserved	PRTN2_COFB1_CLKEN[REQ49]	305	49	0	0	0	No	17
2	Reserved	PRTN2_COFB1_CLKEN[REQ50]	306	50	0	0	0	No	18
2	QuadSPI	PRTN2_COFB1_CLKEN[REQ51]	307	51	1	1	0	No	19
2	Reserved	PRTN2_COFB1_CLKEN[REQ52]	308	52	0	0	0	No	20
2	Reserved	PRTN2_COFB1_CLKEN[REQ53]	309	53	0	0	0	No	21
2	Reserved	PRTN2_COFB1_CLKEN[REQ54]	310	54	0	0	0	No	22
2	SAI_1	PRTN2_COFB1_CLKEN[REQ55]	311	55	1	1	0	No	23
2	Reserved	PRTN2_COFB1_CLKEN[REQ56]	312	56	0	0	0	No	24
2	Reserved	PRTN2_COFB1_CLKEN[REQ57]	313	57	0	0	0	No	25
2	LPCMP_2	PRTN2_COFB1_CLKEN[REQ58]	314	58	1	1	1	No	26
2	HSE_MU1_B	PRTN2_COFB1_CLKEN[REQ59]	315	59	1	0	1	No	27
2	CM7_0_TCM_CLKEN	PRTN2_COFB1_CLKEN[REQ62]	318	62	1	1	1	No	30
2	CM7_1_TCM_CLKEN	PRTN2_COFB1_CLKEN[REQ63]	319	63	1	1	1	No	31

1. See the memory-map sheet attached to this document for details on available peripherals in different chips.
2. While accessing SELFTEST_GPR space, software must ensure that the STCU2 block has its clock enabled. i.e., MC_ME.PRTN1_COFB3_CLKEN[REQ104] configured as 1'b1. Not ensuring this might result in unpredictable device behaviour.

43.1.6 Application core initialization process

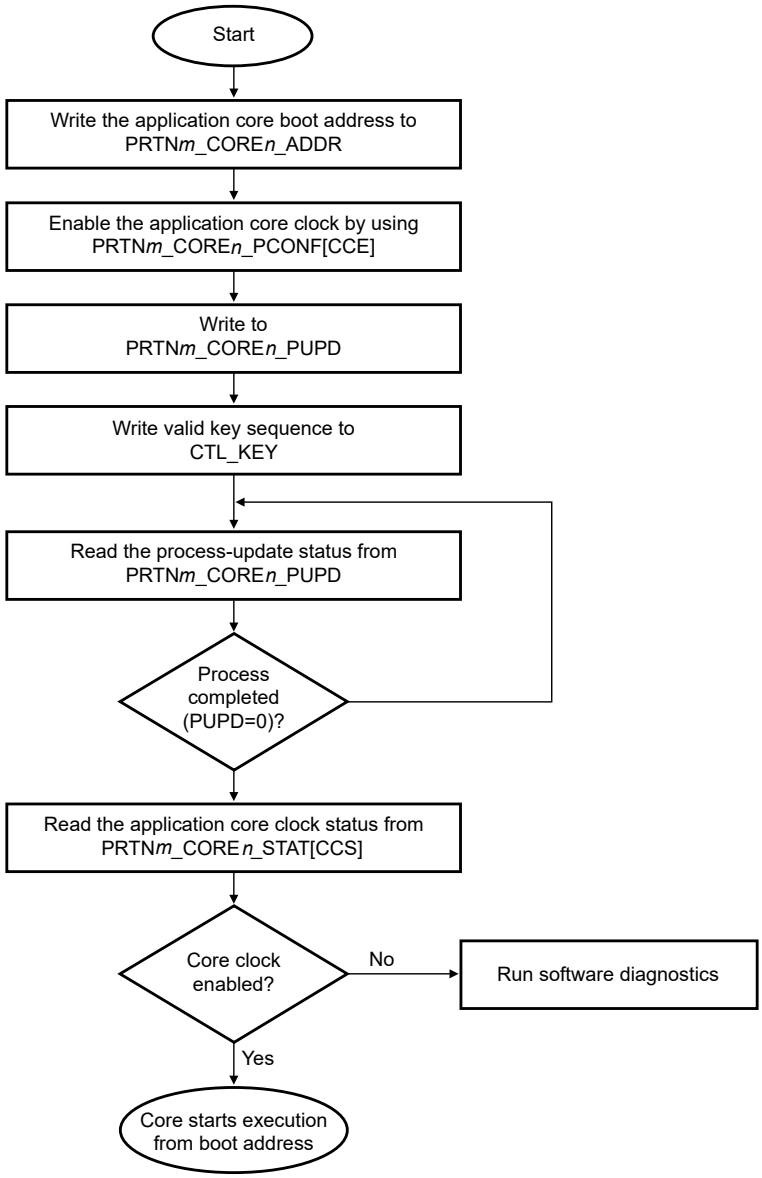


Figure 165. Application core initialization process

43.1.7 Application core shutdown process

If a debugger is attached to the chip and application debug is enabled, the application core continues running if you write 0 to MDM_AP.MDMAPCT[CM7_*n*_CORE_ACCESS].

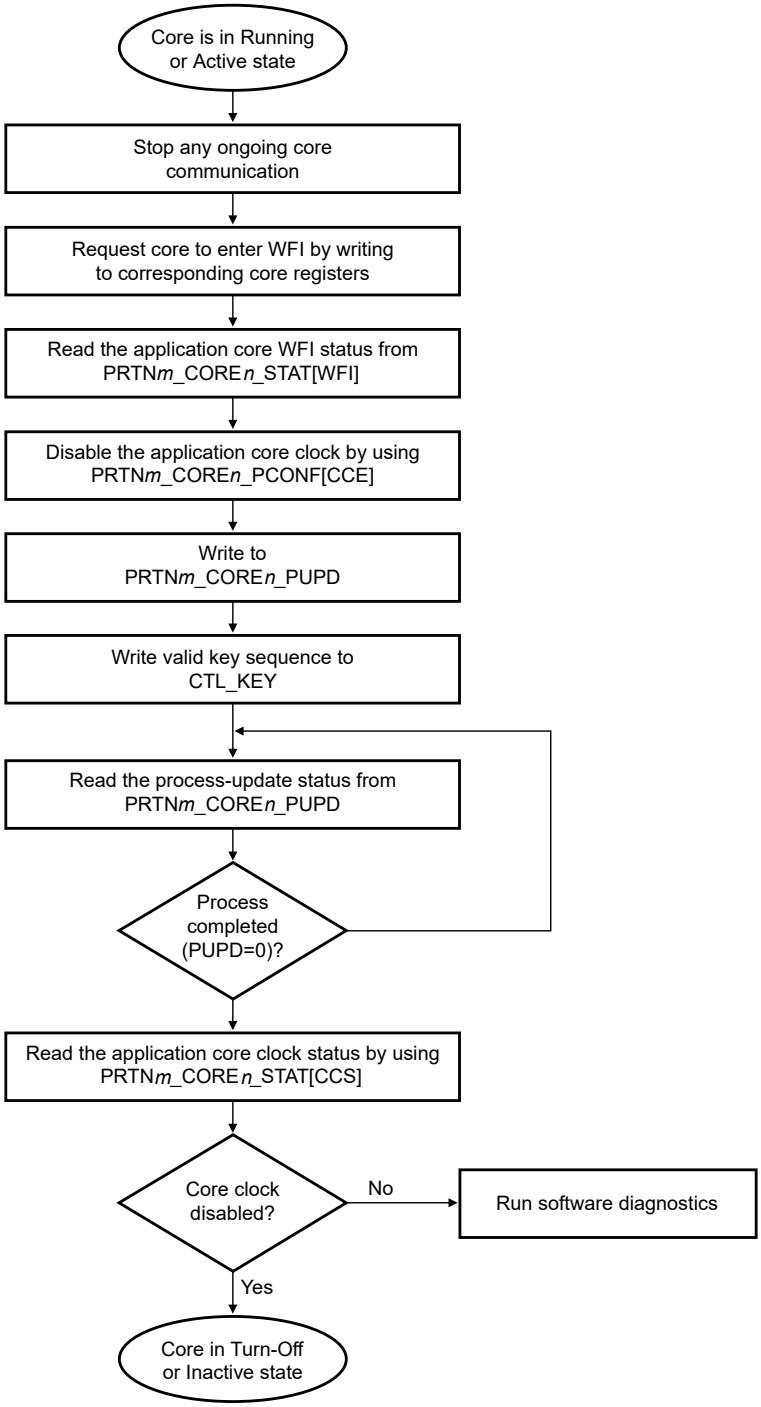


Figure 166. Application core shutdown process

43.1.8 Peripheral initialization process

You cannot control all the peripherals. For example, you cannot turn on and turn off the peripherals required for chip functionality across reset or power-up. They always remain on.

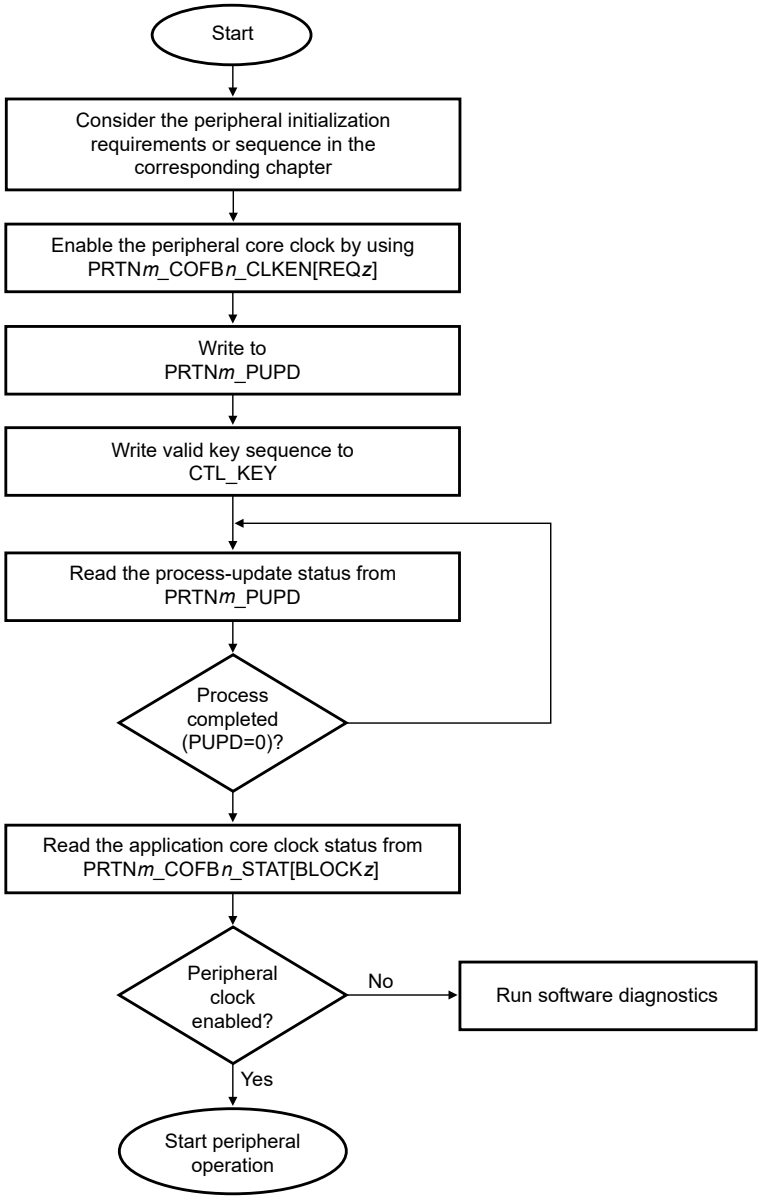


Figure 167. Peripheral initialization process

43.1.9 Peripheral shutdown process

You cannot control all the peripherals. For example, you cannot turn on and turn off the peripherals required for chip functionality across reset or power-up. They always remain on.

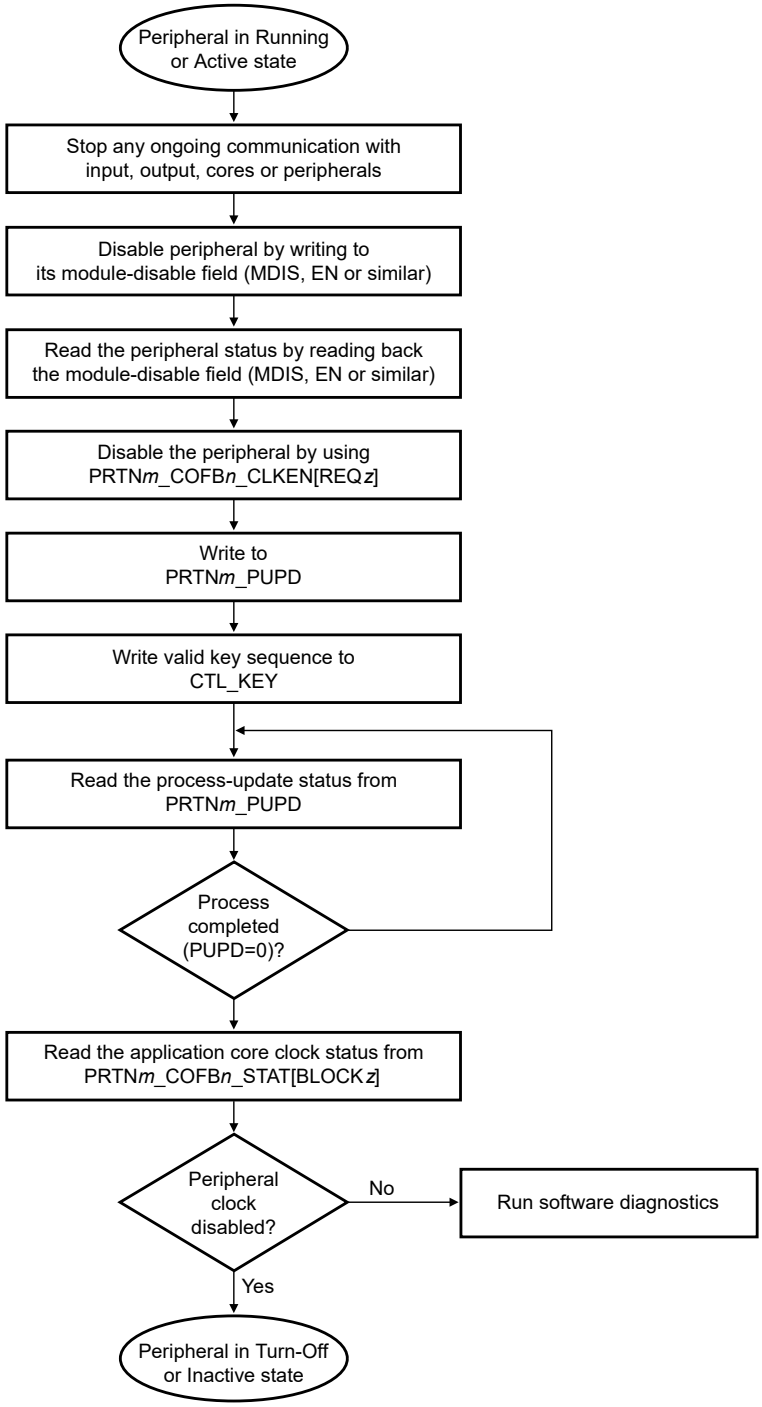


Figure 168. Peripheral shutdown process

43.2 Introduction

The MC_ME module generates control signals for a set of modules of the SoC. The set of signals are defined in corresponding 'Partition Configuration Registers'. It also implements a software-based mechanism for initiating a functional and destructive reset sequence and standby entry handshake with power management of SoC. See Figure 169 for the MC_ME block diagram.

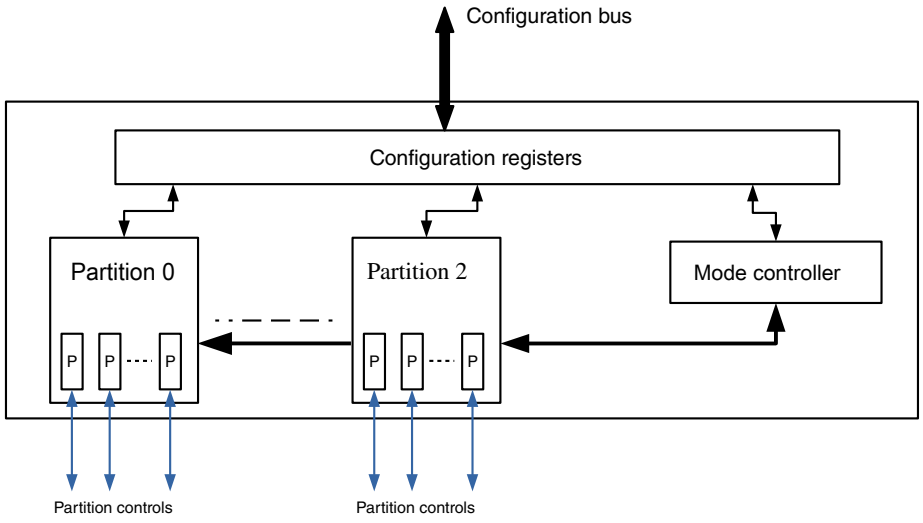


Figure 169. MC_ME block diagram

43.3 Features

MC_ME includes the following features:

- 3 logic partitions implementation and their controls
- Core clock controls
- Partition clock control
- Control mechanism for initiating a destructive or functional reset sequence to MC_RGM
- Control mechanism for initiating standby mode entry for SoC

The logic partition inside MC_ME refers to a certain group of on-chip resources (or IP blocks) that are clubbed together to represent a single 'Partition' inside MC_ME. The MC_ME partition can be the same or different than an LBIST partition. Each of the MC_ME partitions implements a certain number of hardware processes. These hardware processes provide a mechanism to regulate various control signals provided to or received from the IP blocks. The corresponding status signals can also be monitored from MC_ME register(s). Each of the hardware processes is bound to finish in 512 cycles of the MC_ME register configuration clocks. Therefore, the hardware processes are non blocking in nature. Mismatch in the expected versus actual status of any hardware process is controlled by a pre-defined software.

43.4 Partition processes

Each of the processes inside the partition controls register space and corresponds to a control signal provided to that partition. A partition can include a core, or COFBs, or both. The MC_ME hardware processes provide control and status via signals provided to partitions. Each partition can be assigned a signal for control and a signal for status. Each of the control signals implements functionality for the partition. For example, clock gating and peripheral control.

The hardware process can be triggered and monitored using a set of three registers:

- Configuration register; for example, Partition n Process Configuration register
- Update register; for example, Partition n Process Update register
- Status register; for example, Partition n Status register

Similar registers exist for cores inside the partition.

The process setup and triggering procedure is shown in Figure 170. Each of the processes is independent of others and can be triggered or retriggered in parallel or sequential to other processes. The triggering or retriggering mechanism remains the same.

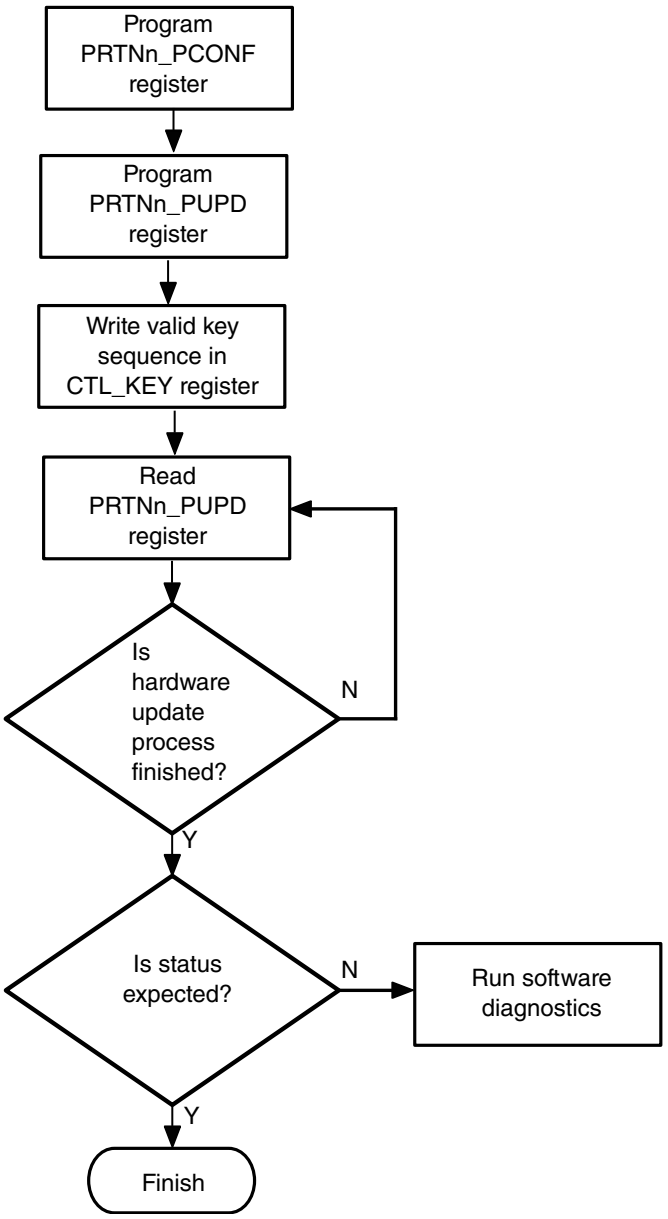


Figure 170. Partition process setup procedure

All the hardware processes are bound to finish in 512 cycles of the MC_ME configuration clock. If the actual and the expected status for a process does not match, then the diagnostics is left as a software responsibility. The software diagnostic can include further wait cycles for the status to match.

43.5 Mode transition

MC_ME implements a mode transition mechanism, whereby the mode of operation for SoC can be changed. Then module implements a mechanism that can lead to:

- Destructive reset
- Functional reset
- Standby mode entry

Destructive reset and functional reset requests from MC_ME are non-retractable transitions. After it is initiated, the other MC_ME functionality is rendered unusable and bus errors are provided for upcoming access to the MC_ME register until a reset sequence is executed by MC_RGM. Hence, it is vital that MC_RGM should never ignore or gate the reset requests from MC_ME.

For transition into the standby mode, the software should ensure that required IP blocks such as clock sources and I/O communication are in their respective inactive states before initiating a standby mode transition to MC_ME. After MC_ME initiates a power down sequence request, it cannot be retracted. The SoC enters a standby power down sequence and then reenter power-up sequence even for cases where the standby wakeup happens right at the time of initiating a power-down sequence request.

Steps for initiating the MC_ME mode transition:

1. Setup the MODE_CONF register with the corresponding target mode bit set to logic-1.
2. Perform the same update as done in the MODE_CONF register on the CONF_UPD register.
3. Write the valid control key (0x5AF0) on the CTL_KEY register.
4. Write the valid invert control key (0xA50F) on the CTL_KEY register.

Mode transition to MC_ME is initiated, after the sequence mentioned above is completed.

In step 1, if both FUNC_RST and DEST_RST in Mode Configuration Register (MODE_CONF) are 1:

- After step 4 is complete, MC_ME initiates a mode transition to a destructive (not functional) reset.
- After the chip exits reset, MC_RGM records that both MC_ME's destructive reset and MC_ME's functional reset were the reset source.

NOTE

Any hardware partition processes setup, along with mode transition, is executed in parallel to the mode transition of MC_ME.

43.6 Standby entry

MC_ME provides hardware processes that implement shutdown sequencing of on-chip resources, such as cores and COFBs. The standby entry sequencing can be achieved or implemented using these hardware processes. The order of the hardware process is determined by the software and MC_ME. It requires no restriction in sequencing of the operation. Following is an example sequence for initiating a power-down sequence for entering the standby mode for SoC. The standby entry sequence should include (but not limited to) the following steps:

1. Setting up wakeup lines
2. Shutting down cores and COFBs
3. Switching all MC_CGM muxes to FIRC with PCFS
4. Powering down all clock sources except FIRC
5. Setting up MC_ME using the main core and initiating a standby mode transition
6. Executing WFI instruction on the main core (per Arm specification)

43.6.1 Application core shutdown

This section describes a mechanism for shutting down an application core. The sequence proposed here is extendible with the housekeeping tasks required for other IPs. Each of the tasks mentioned in the following sequence, can be further integrated with an SoC-specific task.

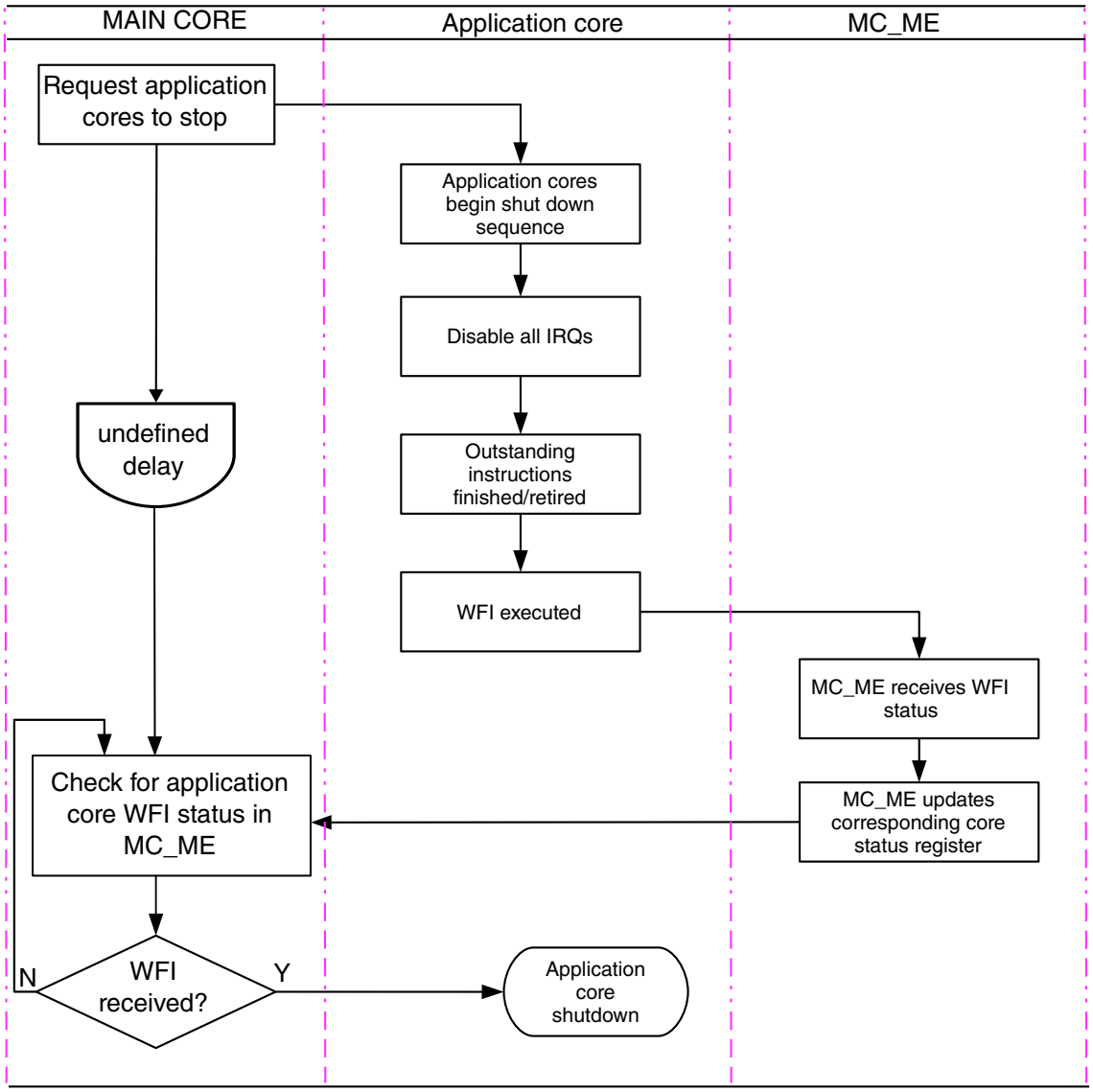


Figure 171. Application core shutdown

After the application core is shutdown, the main core can optionally decide to gate the respective core clock using the corresponding core clock hardware process.

43.6.2 Main core shutdown and standby entry

This section describes standby entry sequence along with the main core shutdown. This sequence should only be initiated after SoC is ready for entering standby and has completed all the housekeeping activities. It is necessary that the main core has completed all the operations pertaining to other (application cores) and is the last active core before initiating the standby entry sequence. See Figure 172.

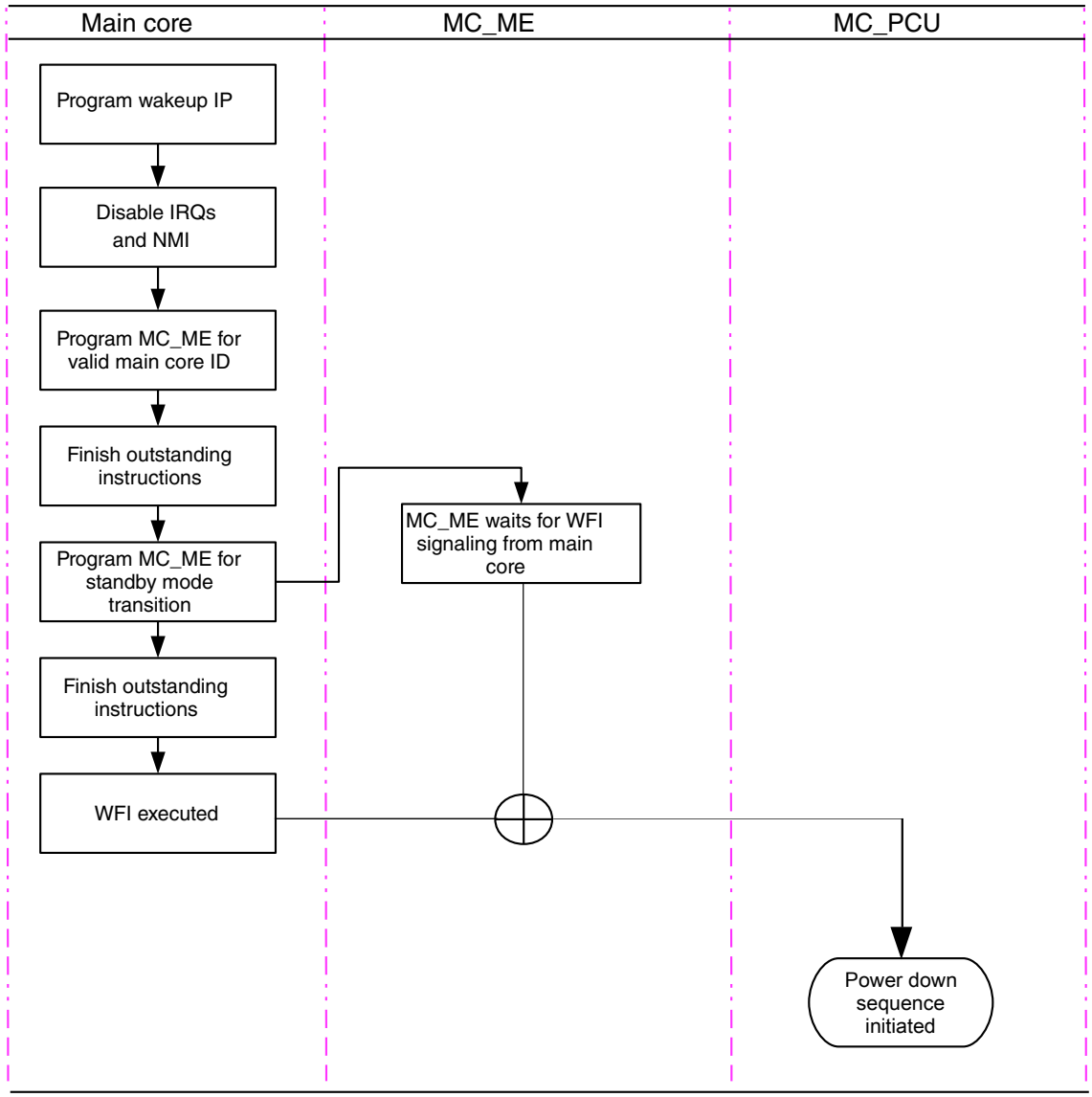


Figure 172. Standby entry sequence along with main core shutdown

NOTE

- MC_ME initiates the power sequence to MC_PCU. This enables the main core to remain inactive (WFI state) until it is reset and power-up again at standby exit.

43.7 MC_ME register descriptions

MC_ME implements set hardware processes that can be used by the software for changing the mode of operation for a partition. Following are the features of MC_ME registers:

- All registers are 32-bit wide.
- Only 32-bit read and write accesses are supported.
- Read/write accesses of less than 32 bits terminate with an error.
- Writes to read-only register fields in writable registers are ignored and do not provide an error message.

- Writes to read-only registers are aborted with an error message.

43.7.1 MC_ME memory map

MC_ME base address: 402D_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Key Register (CTL_KEY)	32	RW	0000_5AF0h
4h	Mode Configuration Register (MODE_CONF)	32	RW	0000_0000h
8h	Mode Update Register (MODE_UPD)	32	RW	0000_0000h
Ch	Mode Status Register (MODE_STAT)	32	R	0000_0000h
10h	Main Core ID Register (MAIN_COREID)	32	RW	0000_0000h
100h	Partition 0 Process Configuration Register (PRTN0_PCONF)	32	RW	0000_0001h
104h	Partition 0 Process Update Register (PRTN0_PUPD)	32	RW	0000_0000h
108h	Partition 0 Status Register (PRTN0_STAT)	32	R	0000_0001h
114h	Partition 0 COFB Set 1 Clock Status Register (PRTN0_COFB1_STAT)	32	R	0000_1000h
134h	Partition 0 COFB Set 1 Clock Enable Register (PRTN0_COFB1_CLKEN)	32	RW	0000_1000h
140h	Partition 0 Core 0 Process Configuration Register (PRTN0_CORE0_PCONF)	32	RW	0000_0000h
144h	Partition 0 Core 0 Process Update Register (PRTN0_CORE0_PUPD)	32	RW	0000_0000h
148h	Partition 0 Core 0 Status Register (PRTN0_CORE0_STAT)	32	R	0000_0000h
14Ch	Partition 0 Core 0 Address Register (PRTN0_CORE0_ADDR)	32	RW	0040_0000h
160h	Partition 0 Core 1 Process Configuration Register (PRTN0_CORE1_PCONF)	32	RW	0000_0000h
164h	Partition 0 Core 1 Process Update Register (PRTN0_CORE1_PUPD)	32	RW	0000_0000h
168h	Partition 0 Core 1 Status Register (PRTN0_CORE1_STAT)	32	R	0000_0000h
16Ch	Partition 0 Core 1 Address Register (PRTN0_CORE1_ADDR)	32	RW	0041_0000h
188h	Partition 0 Core 2 Status Register (PRTN0_CORE2_STAT)	32	R	0000_0001h
18Ch	Partition 0 Core 2 Address Register (PRTN0_CORE2_ADDR)	32	R	007F_FC00h
300h	Partition 1 Process Configuration Register (PRTN1_PCONF)	32	RW	0000_0001h
304h	Partition 1 Process Update Register (PRTN1_PUPD)	32	RW	0000_0000h
308h	Partition 1 Status Register (PRTN1_STAT)	32	R	0000_0001h
310h	Partition 1 COFB Set 0 Clock Status Register (PRTN1_COFB0_STAT)	32	R	5E3F_0007h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
314h	Partition 1 COFB Set 1 Clock Status Register (PRTN1_COFB1_STAT)	32	R	1CFE_2FFCh
318h	Partition 1 COFB Set 2 Clock Status Register (PRTN1_COFB2_STAT)	32	R	3000_0000h
31Ch	Partition 1 COFB Set 3 Clock Status Register (PRTN1_COFB3_STAT)	32	R	0000_5FEEh
330h	Partition 1 COFB Set 0 Clock Enable Register (PRTN1_COFB0_CLKEN)	32	RW	5E3F_0007h
334h	Partition 1 COFB Set 1 Clock Enable Register (PRTN1_COFB1_CLKEN)	32	RW	1CFE_2FFCh
338h	Partition 1 COFB Set 2 Clock Enable Register (PRTN1_COFB2_CLKEN)	32	RW	3000_0000h
33Ch	Partition 1 COFB Set 3 Clock Enable Register (PRTN1_COFB3_CLKEN)	32	RW	0000_5FEEh
500h	Partition 2 Process Configuration Register (PRTN2_PCONF)	32	RW	0000_0001h
504h	Partition 2 Process Update Register (PRTN2_PUPD)	32	RW	0000_0000h
508h	Partition 2 Status Register (PRTN2_STAT)	32	R	0000_0001h
510h	Partition 2 COFB Set 0 Clock Status Register (PRTN2_COFB0_STAT)	32	R	0200_0003h
514h	Partition 2 COFB Set 1 Clock Status Register (PRTN2_COFB1_STAT)	32	R	CC00_0000h
530h	Partition 2 COFB Set 0 Clock Enable Register (PRTN2_COFB0_CLKEN)	32	RW	0200_0003h
534h	Partition 2 COFB Set 1 Clock Enable Register (PRTN2_COFB1_CLKEN)	32	RW	CC00_0000h

43.7.2 Control Key Register (CTL_KEY)

Offset

Register	Offset
CTL_KEY	0h

Function

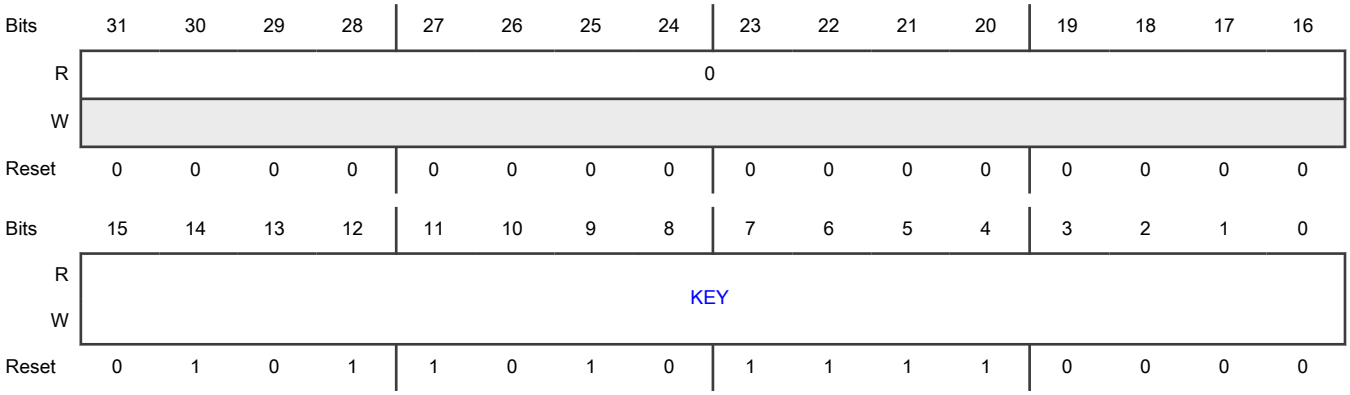
This register provides the mechanism to MC_ME for starting the hardware processes for the partition(s) and standby entry sequence. The hardware processes for partitions are triggered through the corresponding PRTNn_PCONF register. The mechanism to trigger the hardware processes of the respective partitions require two write operations: first time with key and second time with inverted key. The hexadecimal value of key is 0x5AF0 whereas for inverted key is 0xA50F.

For initiating a standby entry sequence, the MODE_CONF register is used for providing a standby entry request along with a valid key combination.

NOTE

Reads from this register return a valid key value to be written next.

Diagram



Fields

Field	Function
31-16 —	Reserved This field is reserved and read returns zeros.
15-0 KEY	Control key Key for starting the hardware processes. Writes with a value other than key or inverted key are ignored. Reads return bit inverted value corresponding to last write.

43.7.3 Mode Configuration Register (MODE_CONF)

Offset

Register	Offset
MODE_CONF	4h

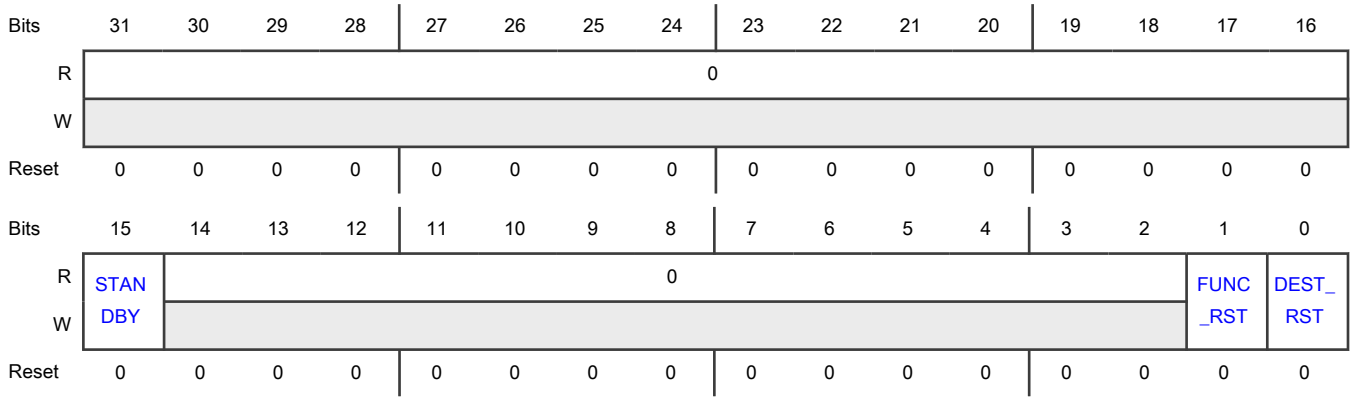
Function

This register is used for initiating a standby request or a reset event (destructive or functional) for the chip. The functional or destructive events are signaled to MC_RGM for further handling.

NOTE

Software must not enable mode entry if the value of multiple fields is 1 in the MODE_CONF register.

Diagram



Fields

Field	Function
31-16 —	Reserved This field is reserved and read returns zeros.
15 STANDBY	Standby request Writing a logic-1 to this bit along with the MODE_UPD register configuration and followed with a valid key combination makes a standby entry sequence request to MC_ME.
14-2 —	Reserved This field is reserved and read returns zeros.
1 FUNC_RST	Functional reset request Writing a logic-1 to this bit along with the MODE_UPD register configuration and followed with a valid key combination makes a functional reset event signaling to MC_RGM.
0 DEST_RST	Destructive reset request Writing a logic-1 to this bit along with the MODE_UPD register configuration and followed with a valid key combination makes a destructive reset event signaling to MC_RGM.

43.7.4 Mode Update Register (MODE_UPD)

Offset

Register	Offset
MODE_UPD	8h

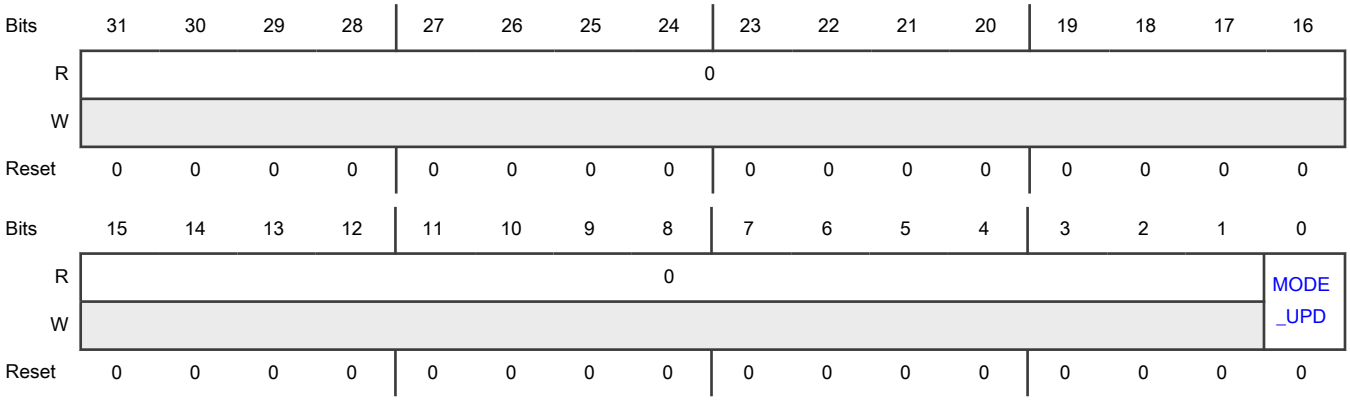
Function

This register is used for initiating a mode change. Mode change refers to initiating a standby request, or generating a destructive or functional reset event to MC_RGM. Setting mode update field to logic-1, along with programming MODE_CONF registers and then followed by a valid key combination will generate a mode transition request.

NOTE

The MODE_UPD register is implemented to make mode transition programming model the same as partition programming model. This is for future expansion inside MC_ME.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 MODE_UPD	Mode update Writing a logic-1 to this bit, followed by a valid key combination initiates a mode change as per the MODE_CONF register.

43.7.5 Mode Status Register (MODE_STAT)

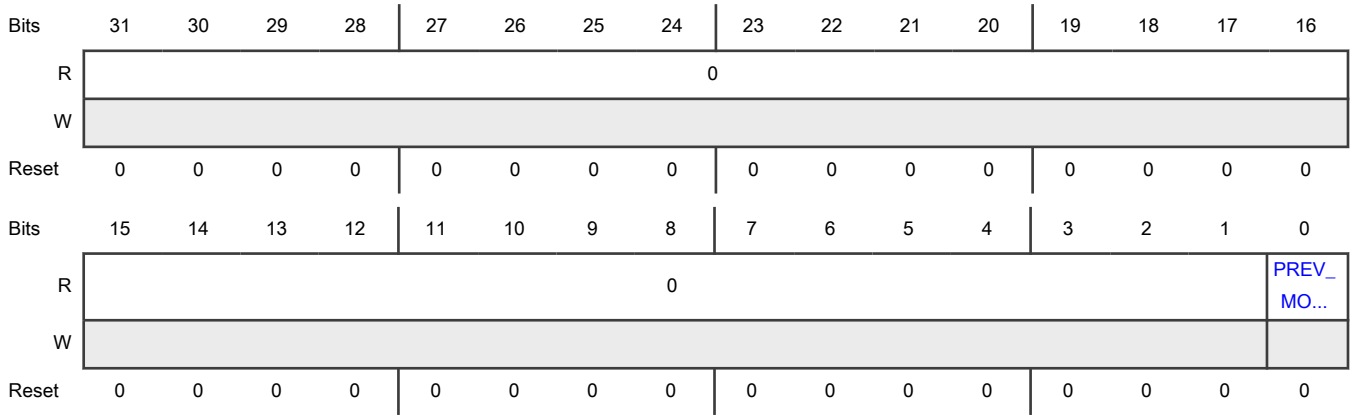
Offset

Register	Offset
MODE_STAT	Ch

Function

This register provides the status of the previous mode. In case of standby exit, if the reset event status register of MC_RGM are set, then contents of this register should be ignored.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 PREV_MODE	Previous mode This bit shows the status of the previous mode. 0b - The previous mode was reset (any reset). 1b - The previous mode was standby.

43.7.6 Main Core ID Register (MAIN_COREID)

Offset

Register	Offset
MAIN_COREID	10h

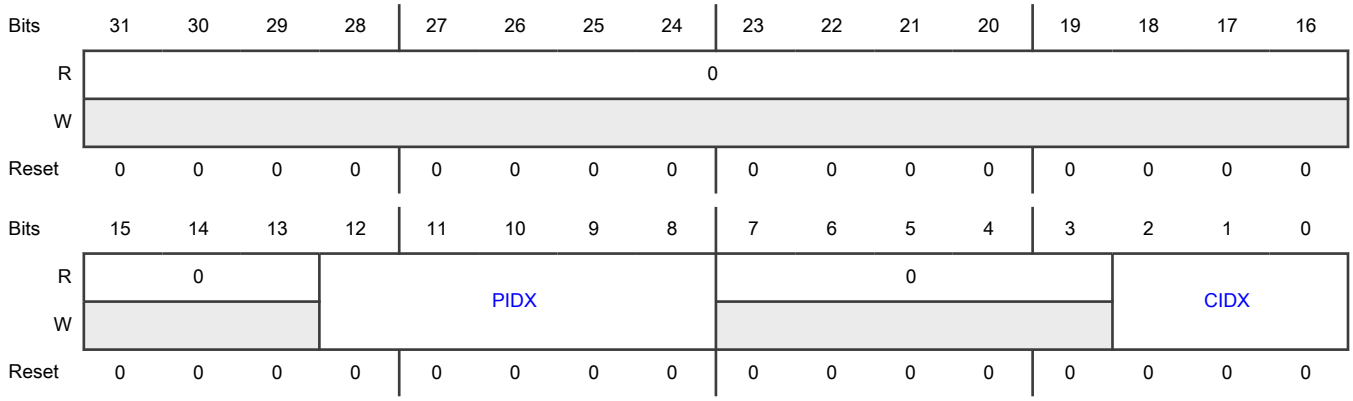
Function

This register provides the ID of the main core sequencing the operation for the standby sequence. Core ID is required for entering in the standby mode, and using this MC_ME locates the WFI instruction execution of the main core. The core ID in this register is specified by the partition index along with the core index.

NOTE

Before initiating a standby entry sequence, the contents of this register should point to the correct main core. Providing non-existing or incorrect core ID leads to unpredictable hardware behavior.

Diagram



Fields

Field	Function
31-13 —	Reserved This field is reserved and read returns zeros.
12-8 PIDX	Partition index Provides the partition index of the main core. Only values 0 - 2 can be written.
7-3 —	Reserved This field is reserved and read returns zeros.
2-0 CIDX	Core index Provides the core index of the main core inside the partition.

43.7.7 Partition 0 Process Configuration Register (PRTN0_PCONF)

Offset

Register	Offset
PRTN0_PCONF	100h

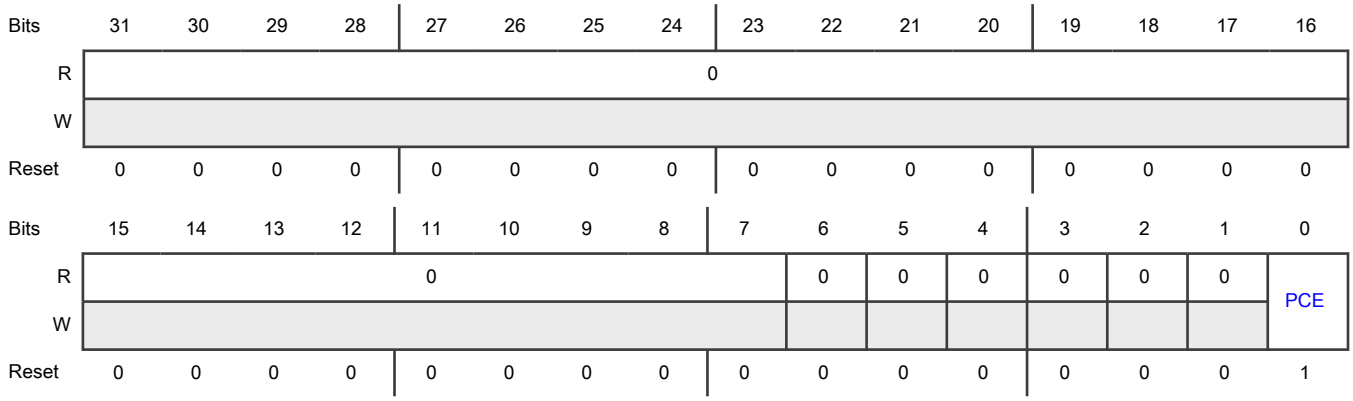
Function

This register provides a configuration for the hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_PCONF and PRTN0_PUPD registers are used to determine the hardware processes to be executed. These are triggered in parallel and independent of each other. All dependent processes should be requested one after another from the software.

NOTE

The partition clock enable/disable and output safe stating enable/disable are not standalone and must be done coherently in a fixed sequence. For details, see Software Reset Partition Turn-On Flow Chart and Software reset partition turn-off flowchart in Reset chapter.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCE	Partition clock enable This bit controls whether the clock to IPs (other than core(s)) in the partition should be enabled or disabled. 0b - Disable the clock to IPs 1b - Enable the clock to IPs

43.7.8 Partition 0 Process Update Register (PRTN0_PUPD)

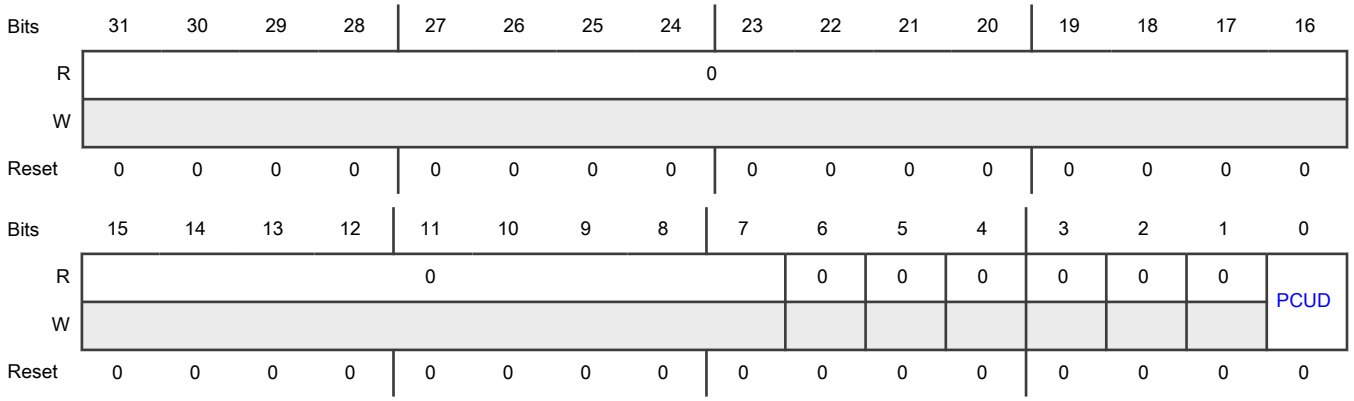
Offset

Register	Offset
PRTN0_PUPD	104h

Function

This register provides trigger signaling for the hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_PCONF register. When the hardware process is finished the corresponding bit in this register is auto-cleared to logic-0.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCUD	Partition clock update This bit controls whether the hardware processes for enabling/disabling the clock to IPs (other than core(s)) in the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

43.7.9 Partition 0 Status Register (PRTN0_STAT)

Offset

Register	Offset
PRTN0_STAT	108h

Function

This register provides the current status of the control signals from the partition 0.

Diagram

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

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

Fields

Field	Function
31-7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCS	Partition clock status This bit provides the status of the clock to partition. 0b - Clock is inactive 1b - Clock is active

43.7.10 Partition 0 COFB Set 1 Clock Status Register (PRTN0_COFB1_STAT)

Offset

Register	Offset
PRTN0_COFB1_STAT	114h

Function

This register provides the status of set 1 of COFBs inside partition 0.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 0.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK47	BLOCK46	BLOCK45	BLOCK44	0	BLOCK42	BLOCK41	BLOCK40	BLOCK39	BLOCK38	0	BLOCK36	BLOCK35	BLOCK34	BLOCK33	BLOCK32
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved This field is reserved and read returns zeros.
30 —	Reserved This field is reserved and read returns zeros.
29 —	Reserved This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 —	Reserved This field is reserved and read returns zeros.
26 —	Reserved This field is reserved and read returns zeros.
25 —	Reserved This field is reserved and read returns zeros.
24 —	Reserved This field is reserved and read returns zeros.
23 —	Reserved This field is reserved and read returns zeros.
22 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved This field is reserved and read returns zeros.
20 —	Reserved This field is reserved and read returns zeros.
19 —	Reserved This field is reserved and read returns zeros.
18 —	Reserved This field is reserved and read returns zeros.
17 —	Reserved This field is reserved and read returns zeros.
16 —	Reserved This field is reserved and read returns zeros.
15 BLOCK47	IP block status This bit provides the clock status of block 47 in partition 0. 0b - Clock is not running. 1b - Clock is running.
14 BLOCK46	IP block status This bit provides the clock status of block 46 in partition 0. 0b - Clock is not running. 1b - Clock is running.
13 BLOCK45	IP block status This bit provides the clock status of PIT_1 in partition 0. 0b - Clock is not running. 1b - Clock is running.
12 BLOCK44	IP block status This bit provides the clock status of PIT_0 in partition 0. 0b - Clock is not running. 1b - Clock is running.
11 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 BLOCK42	IP block status This bit provides the clock status of ADC_2 in partition 0. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK41	IP block status This bit provides the clock status of ADC_1 in partition 0. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK40	IP block status This bit provides the clock status of ADC_0 in partition 0. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK39	IP block status This bit provides the clock status of LCU_1 in partition 0. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK38	IP block status This bit provides the clock status of LCU_0 in partition 0. 0b - Clock is not running. 1b - Clock is running.
5 —	Reserved This field is reserved and read returns zeros.
4 BLOCK36	IP block status This bit provides the clock status of eMIOS_2 in partition 0. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK35	IP block status This bit provides the clock status of eMIOS_1 in partition 0. 0b - Clock is not running. 1b - Clock is running.
2	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK34	This bit provides the clock status of eMIOS_0 in partition 0. 0b - Clock is not running. 1b - Clock is running.
1 BLOCK33	IP block status This bit provides the clock status of BCTU in partition 0. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK32	IP block status This bit provides the clock status of TRGMUX in partition 0. 0b - Clock is not running. 1b - Clock is running.

43.7.11 Partition 0 COFB Set 1 Clock Enable Register (PRTN0_COFB1_CLKEN)

Offset

Register	Offset
PRTN0_COFB1_CLKEN	134h

Function

This register provides clock control signaling to the individual COFBs in set 1 inside partition 0. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ47	REQ46	REQ45	REQ44	0	REQ42	REQ41	REQ40	REQ39	REQ38	0	REQ36	REQ35	REQ34	REQ33	REQ32
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved This field is reserved and read returns zeros.
30 —	Reserved This field is reserved and read returns zeros.
29 —	Reserved This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 —	Reserved This field is reserved and read returns zeros.
26 —	Reserved This field is reserved and read returns zeros.
25 —	Reserved This field is reserved and read returns zeros.
24 —	Reserved This field is reserved and read returns zeros.
23 —	Reserved This field is reserved and read returns zeros.
22 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved This field is reserved and read returns zeros.
20 —	Reserved This field is reserved and read returns zeros.
19 —	Reserved This field is reserved and read returns zeros.
18 —	Reserved This field is reserved and read returns zeros.
17 —	Reserved This field is reserved and read returns zeros.
16 —	Reserved This field is reserved and read returns zeros.
15 REQ47	Clock enable This bit provides the clock enable control for block 47 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
14 REQ46	Clock enable This bit provides the clock enable control for block 46 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
13 REQ45	Clock enable This bit provides the clock enable control for PIT_1 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
12 REQ44	Clock enable This bit provides the clock enable control for PIT_0 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
11 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 REQ42	Clock enable This bit provides the clock enable control for ADC_2 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ41	Clock enable This bit provides the clock enable control for ADC_1 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
8 REQ40	Clock enable This bit provides the clock enable control for ADC_0 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
7 REQ39	Clock enable This bit provides the clock enable control for LCU_1 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
6 REQ38	Clock enable This bit provides the clock enable control for LCU_0 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
5 —	Reserved This field is reserved and read returns zeros.
4 REQ36	Clock enable This bit provides the clock enable control for eMIOS_2 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
3 REQ35	Clock enable This bit provides the clock enable control for eMIOS_1 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
2	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ34	This bit provides the clock enable control for eMIOS_0 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
1 REQ33	Clock enable This bit provides the clock enable control for BCTU in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
0 REQ32	Clock enable This bit provides the clock enable control for TRGMUX in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.

43.7.12 Partition 0 Core 0 Process Configuration Register (PRTN0_CORE0_PCONF)

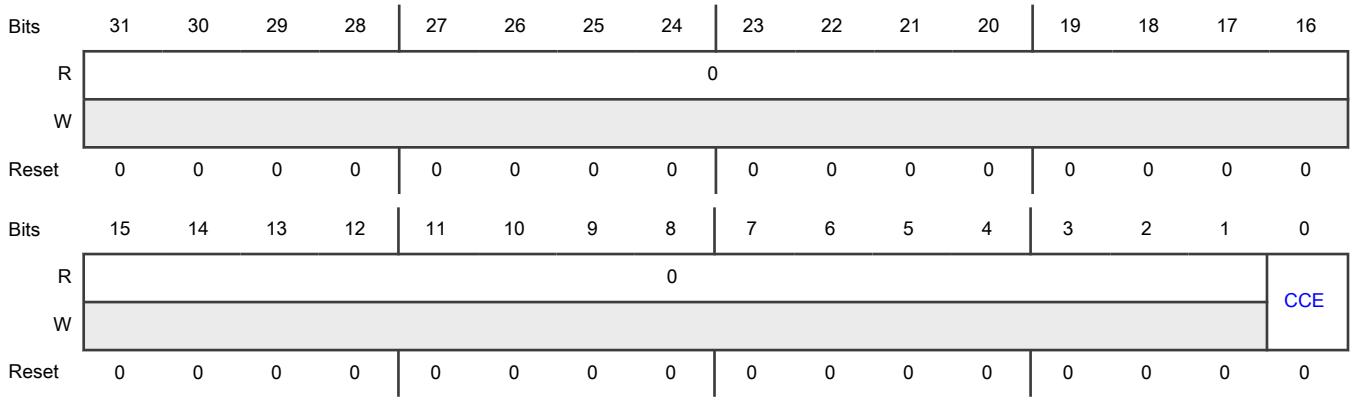
Offset

Register	Offset
PRTN0_CORE0_PCONF	140h

Function

This register provides configurations for the Core 0 hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_CORE0_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_CORE0_PUPD and PRTN0_CORE0_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 0 clock enable This bit controls whether the clock to Core 0 in partition 0 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

43.7.13 Partition 0 Core 0 Process Update Register (PRTN0_CORE0_PUPD)

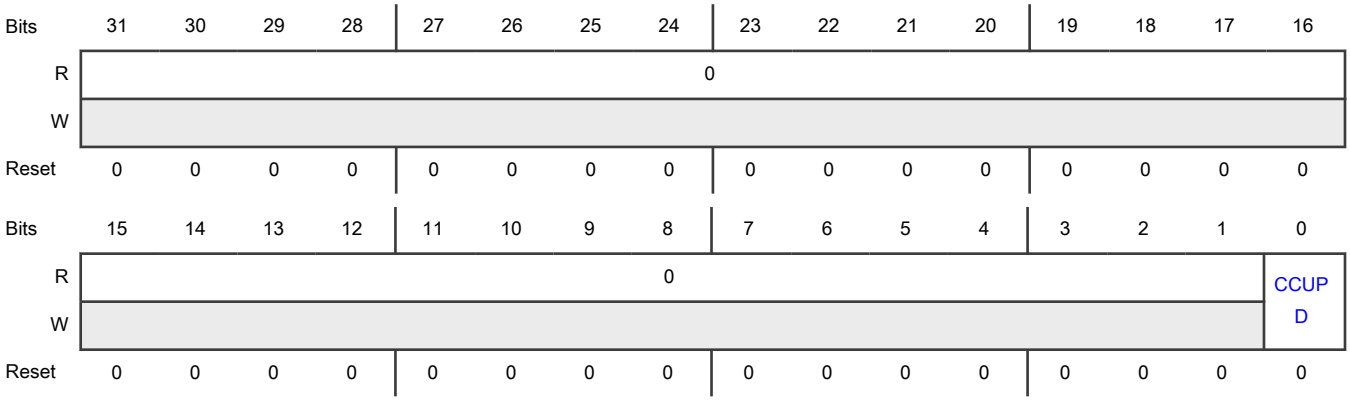
Offset

Register	Offset
PRTN0_CORE0_PUPD	144h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_CORE0_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Diagram



Fields

Field	Function
31-1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
0 CCUPD	Core 0 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 0 in the partition 0 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

43.7.14 Partition 0 Core 0 Status Register (PRTN0_CORE0_STAT)

Offset

Register	Offset
PRTN0_CORE0_STAT	148h

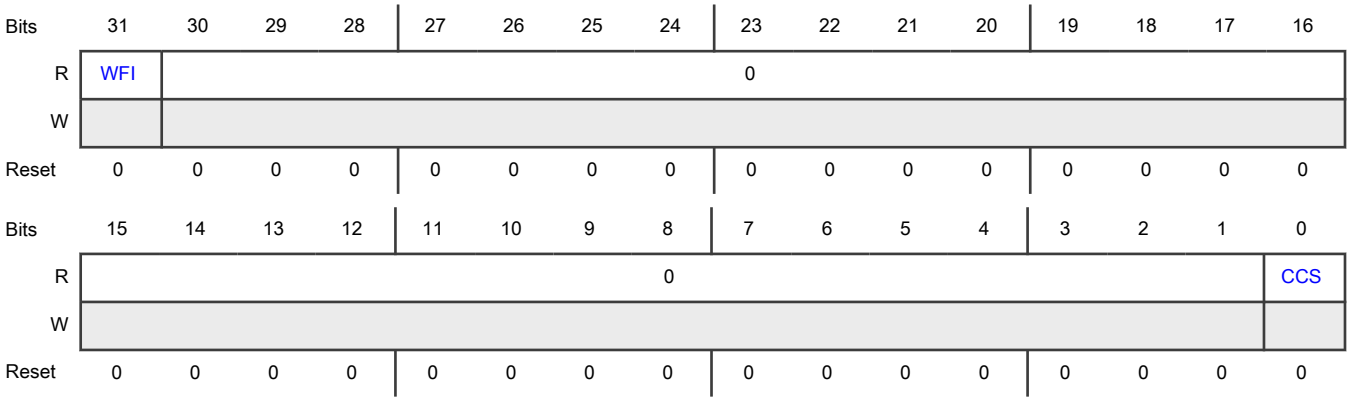
Function

This register provides the status corresponding to Core 0 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 0.

NOTE

The value held in WFI field of this STATUS register is "current" value of the WFISTANDBY signal from the core. Hence out-of-reset, the reset value of this field will depend on the status of the core (core is running or in low power mode). So, simple reset read sweep will always return current value (different than other register reads such as on control registers).

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 0 in partition 0. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 0 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

43.7.15 Partition 0 Core 0 Address Register (PRTN0_CORE0_ADDR)

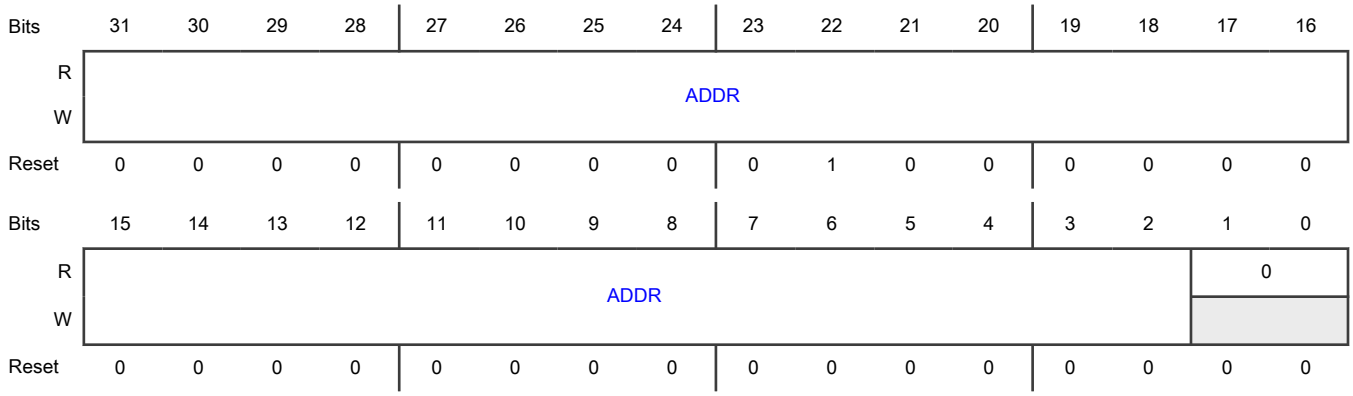
Offset

Register	Offset
PRTN0_CORE0_ADDR	14Ch

Function

This register contains the boot address for Core 0 in partition 0.

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 0 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

43.7.16 Partition 0 Core 1 Process Configuration Register (PRTN0_CORE1_PCONF)

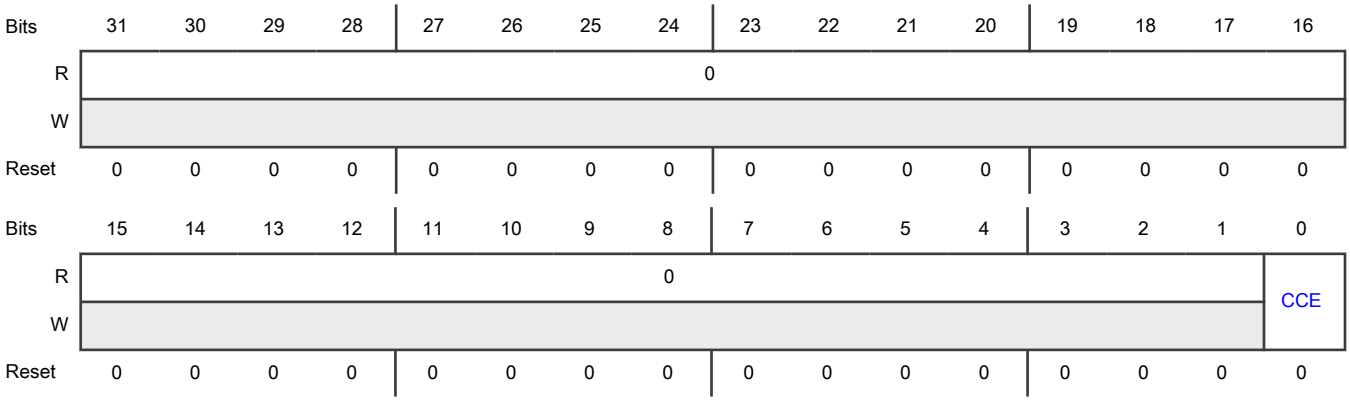
Offset

Register	Offset
PRTN0_CORE1_PCONF	160h

Function

This register provides configurations for the Core 1 hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_CORE1_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_CORE1_PUPD and PRTN0_CORE1_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 CCE	Core 1 clock enable This bit controls whether the clock to Core 1 in partition 0 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

43.7.17 Partition 0 Core 1 Process Update Register (PRTN0_CORE1_PUPD)

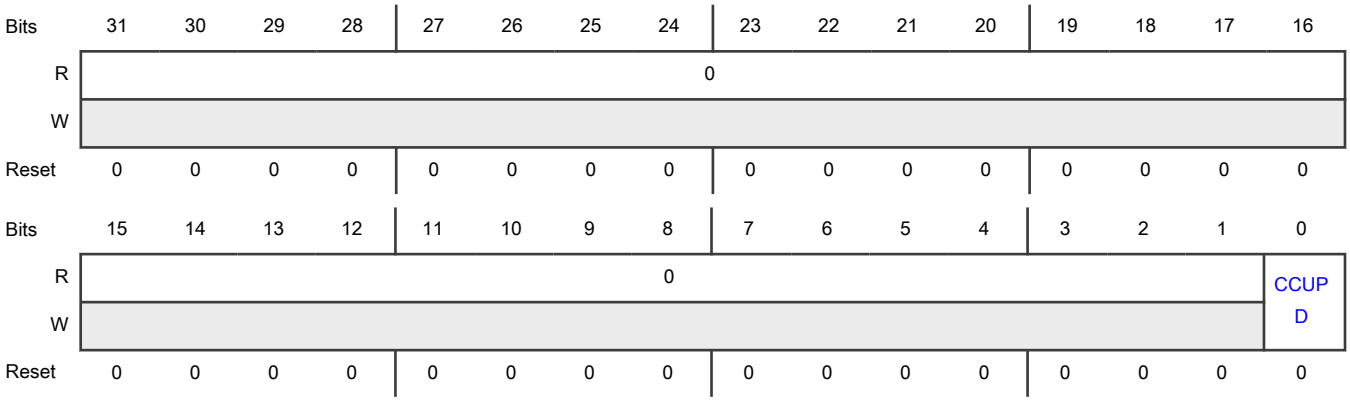
Offset

Register	Offset
PRTN0_CORE1_PUPD	164h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_CORE1_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0	Core 1 clock update

Table continues on the next page...

Table continued from the previous page...

Field	Function
CCUPD	This bit controls whether the hardware processes for enabling/disabling the clock to Core 1 in the partition 0 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

43.7.18 Partition 0 Core 1 Status Register (PRTN0_CORE1_STAT)

Offset

Register	Offset
PRTN0_CORE1_STAT	168h

Function

This register provides the status corresponding to Core 1 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 1.

NOTE

The value held in WFI field of this STATUS register is "current" value of the WFISTANDBY signal from the core. Hence out-of-reset, the reset value of this field will depend on the status of the core (core is running or in low power mode). So, simple reset read sweep will always return current value (different than other register reads such as on control registers).

Diagram

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

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

Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 1 in partition 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 1 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

43.7.19 Partition 0 Core 1 Address Register (PRTN0_CORE1_ADDR)

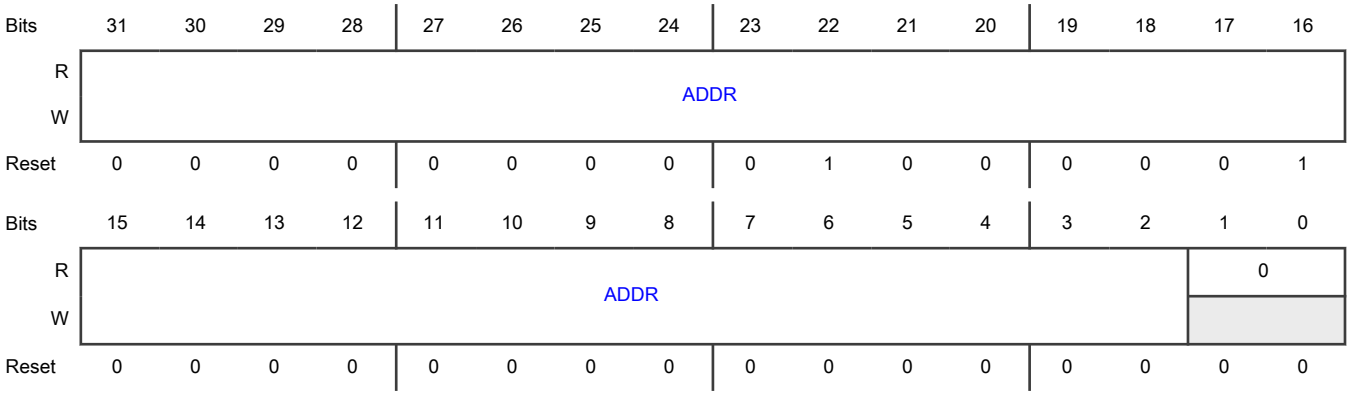
Offset

Register	Offset
PRTN0_CORE1_ADDR	16Ch

Function

This register contains the boot address for Core 1 in partition 0.

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 1 boot address

Table continues on the next page...

Table continued from the previous page...

Field	Function
1-0 —	Reserved This field is reserved and read returns zeros.

43.7.20 Partition 0 Core 2 Status Register (PRTN0_CORE2_STAT)

Offset

Register	Offset
PRTN0_CORE2_STAT	188h

Function

This register provides the status corresponding to Core 2 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 2.

NOTE

The value held in WFI field of this STATUS register is "current" value of the WFISTANDBY signal from the core. Hence out-of-reset, the reset value of this field will depend on the status of the core (core is running or in low power mode). So, simple reset read sweep will always return current value (different than other register reads such as on control registers).

Diagram

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

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

Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 2 in partition 0. 0b - No WFI executed

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 2 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 1b - Clock is active.

43.7.21 Partition 0 Core 2 Address Register (PRTN0_CORE2_ADDR)

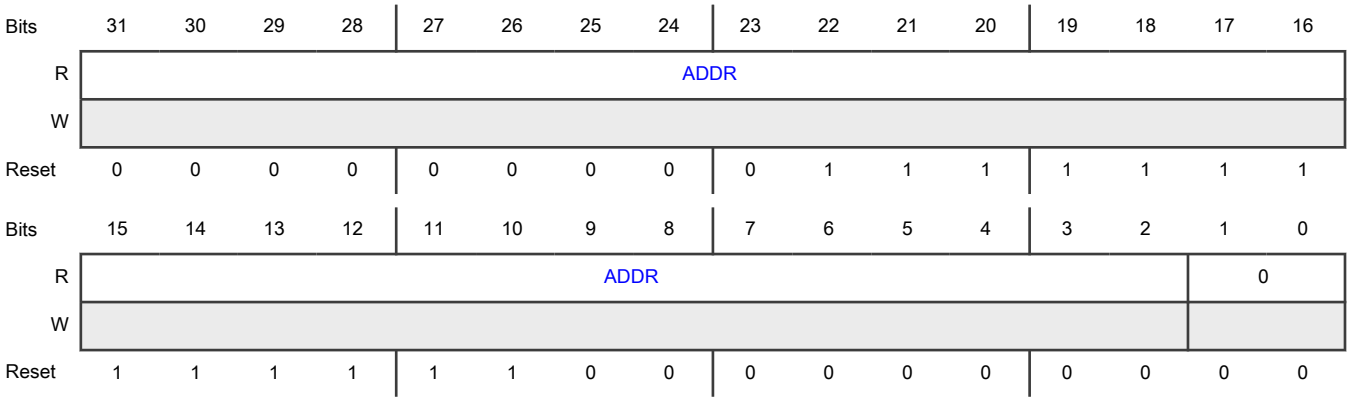
Offset

Register	Offset
PRTN0_CORE2_ADDR	18Ch

Function

This register contains the boot address for Core 2 in partition 0.

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 2 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

43.7.22 Partition 1 Process Configuration Register (PRTN1_PCONF)

Offset

Register	Offset
PRTN1_PCONF	300h

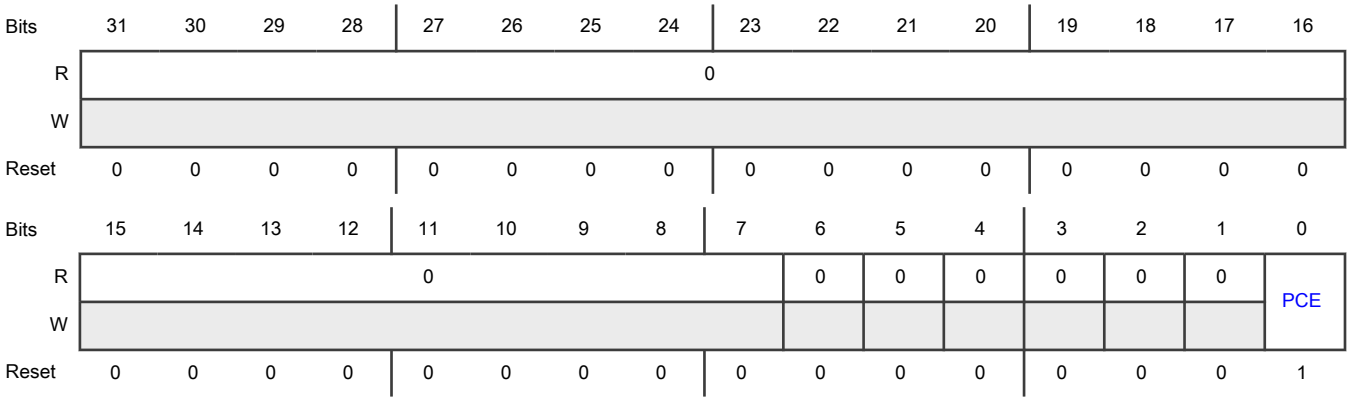
Function

This register provides a configuration for the hardware processes corresponding to partition 1. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN1_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN1_PCONF and PRTN1_PUPD registers are used to determine the hardware processes to be executed. These are triggered in parallel and independent of each other. All dependent processes should be requested one after another from the software.

NOTE

The partition clock enable/disable and output safe stating enable/disable are not standalone and must be done coherently in a fixed sequence. For details, see Software Reset Partition Turn-On Flow Chart and Software reset partition turn-off flowchart in Reset chapter.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCE	Partition clock enable This bit controls whether the clock to IPs (other than core(s)) in the partition should be enabled or disabled. 0b - Disable the clock to IPs 1b - Enable the clock to IPs

43.7.23 Partition 1 Process Update Register (PRTN1_PUPD)

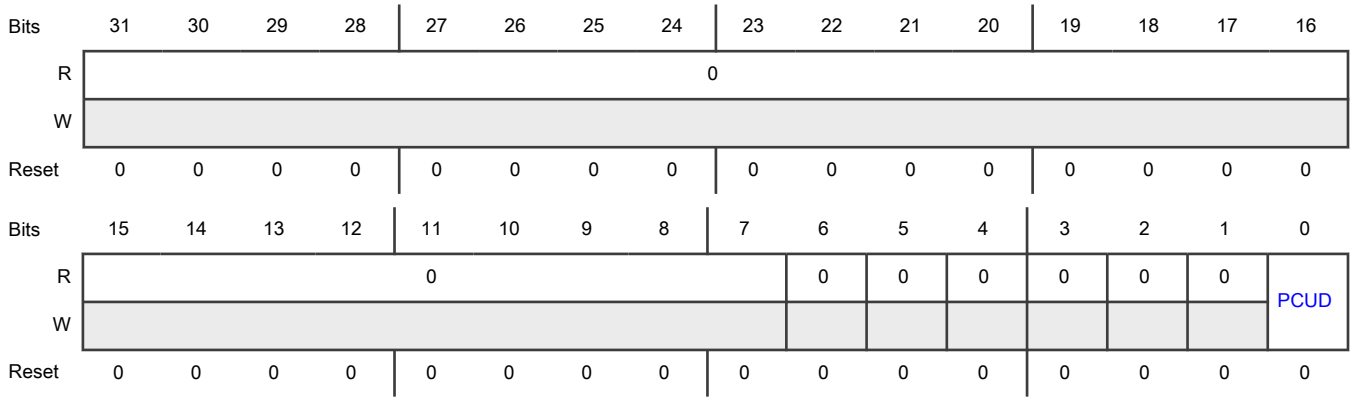
Offset

Register	Offset
PRTN1_PUPD	304h

Function

This register provides trigger signaling for the hardware processes corresponding to partition 1. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN1_PCONF register. When the hardware process is finished the corresponding bit in this register is auto-cleared to logic-0.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCUD	Partition clock update This bit controls whether the hardware processes for enabling/disabling the clock to IPs (other than core(s)) in the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

43.7.24 Partition 1 Status Register (PRTN1_STAT)

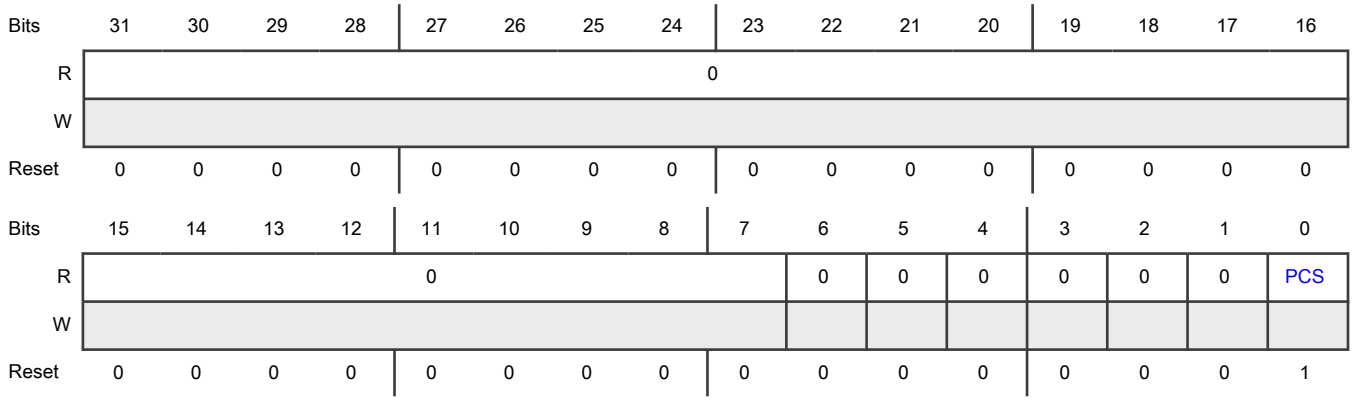
Offset

Register	Offset
PRTN1_STAT	308h

Function

This register provides the current status of the control signals from the partition 1.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCS	Partition clock status This bit provides the status of the clock to partition. 0b - Clock is inactive 1b - Clock is active

43.7.25 Partition 1 COFB Set 0 Clock Status Register (PRTN1_COFB0_STAT)

Offset

Register	Offset
PRTN1_COFB0_STAT	310h

Function

This register provides the status of set 0 of COFBs inside partition 1.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK31	BLOCK30	BLOCK29	BLOCK28	BLOCK27	BLOCK26	BLOCK25	BLOCK24	BLOCK23	BLOCK22	BLOCK21	BLOCK20	BLOCK19	BLOCK18	BLOCK17	BLOCK16
W																
Reset	0	1	0	1	1	1	1	0	0	0	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK15	BLOCK14	BLOCK13	BLOCK12	BLOCK11	BLOCK10	BLOCK9	BLOCK8	BLOCK7	BLOCK6	BLOCK5	BLOCK4	BLOCK3	BLOCK2	BLOCK1	BLOCK0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31 BLOCK31	IP block status This bit provides the clock status of INTM in partition 1. 0b - Clock is not running. 1b - Clock is running.
30 BLOCK30	IP block status This bit provides the clock status of XRDC in partition 1. 0b - Clock is not running. 1b - Clock is running.
29 BLOCK29	IP block status This bit provides the clock status of STM_0 in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
28 BLOCK28	IP block status This bit provides the clock status of SWT_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK27	IP block status This bit provides the clock status of PFC_alt in partition 1. 0b - Clock is not running. 1b - Clock is running.
26 BLOCK26	IP block status This bit provides the clock status of PFC in partition 1. 0b - Clock is not running. 1b - Clock is running.
25 BLOCK25	IP block status This bit provides the clock status of PRAM_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
24 BLOCK24	IP block status This bit provides the clock status of MSCM in partition 1. 0b - Clock is not running. 1b - Clock is running.
23 BLOCK23	IP block status This bit provides the clock status of ERM in partition 1. 0b - Clock is not running. 1b - Clock is running.
22 BLOCK22	IP block status This bit provides the clock status of EIM in partition 1. 0b - Clock is not running. 1b - Clock is running.
21	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK21	This bit provides the clock status of SDA-AP in partition 1. 0b - Clock is not running. 1b - Clock is running.
20 BLOCK20	IP block status This bit provides the clock status of Debug_APB in partition 1. 0b - Clock is not running. 1b - Clock is running.
19 BLOCK19	IP block status This bit provides the clock status of Debug_APB in partition 1. 0b - Clock is not running. 1b - Clock is running.
18 BLOCK18	IP block status This bit provides the clock status of Debug_APB in partition 1. 0b - Clock is not running. 1b - Clock is running.
17 BLOCK17	IP block status This bit provides the clock status of Debug_APB in partition 1. 0b - Clock is not running. 1b - Clock is running.
16 BLOCK16	IP block status This bit provides the clock status of Debug_APB in partition 1. 0b - Clock is not running. 1b - Clock is running.
15 BLOCK15	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running. 1b - Clock is running.
14 BLOCK14	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 BLOCK13	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running. 1b - Clock is running.
12 BLOCK12	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running. 1b - Clock is running.
11 BLOCK11	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running. 1b - Clock is running.
10 BLOCK10	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK9	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK8	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK7	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK6	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock is running.
5 BLOCK5	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running. 1b - Clock is running.
4 BLOCK4	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK3	IP block status This bit provides the clock status of eDMA in partition 1. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK2	IP block status This bit provides the clock status of XBIC in partition 1. 0b - Clock is not running. 1b - Clock is running.
1 BLOCK1	IP block status This bit provides the clock status of XBIC in partition 1. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK0	IP block status This bit provides the clock status of AXBS in partition 1. 0b - Clock is not running. 1b - Clock is running.

43.7.26 Partition 1 COFB Set 1 Clock Status Register (PRTN1_COFB1_STAT)

Offset

Register	Offset
PRTN1_COFB1_STAT	314h

Function

This register provides the status of set 1 of COFBs inside partition 1.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK63	0	0	BLOCK60	BLOCK59	BLOCK58	0	BLOCK56	BLOCK55	BLOCK54	BLOCK53	BLOCK52	BLOCK51	BLOCK50	BLOCK49	0
W																
Reset	0	0	0	1	1	1	0	0	1	1	1	1	1	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK47	0	BLOCK45	0	BLOCK43	BLOCK42	BLOCK41	BLOCK40	BLOCK39	BLOCK38	BLOCK37	BLOCK36	BLOCK35	BLOCK34	BLOCK33	BLOCK32
W																
Reset	0	0	1	0	1	1	1	1	1	1	1	1	1	1	0	0

Fields

Field	Function
31 BLOCK63	IP block status This bit provides the clock status of PIT_2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
30 —	Reserved This field is reserved and read returns zeros.
29 —	Reserved This field is reserved and read returns zeros.
28 BLOCK60	IP block status This bit provides the clock status of FMU_alt in partition 1. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK59	IP block status This bit provides the clock status of FMU in partition 1. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 BLOCK58	IP block status This bit provides the clock status of PMC in partition 1. 0b - Clock is not running. 1b - Clock is running.
25 —	Reserved This field is reserved and read returns zeros.
24 BLOCK56	IP block status This bit provides the clock status of PLL in partition 1. 0b - Clock is not running. 1b - Clock is running.
23 BLOCK55	IP block status This bit provides the clock status of MC_ME in partition 1. 0b - Clock is not running. 1b - Clock is running.
22 BLOCK54	IP block status This bit provides the clock status of MC_CGM in partition 1. 0b - Clock is not running. 1b - Clock is running.
21 BLOCK53	IP block status This bit provides the clock status of FXOSC in partition 1. 0b - Clock is not running. 1b - Clock is running.
20 BLOCK52	IP block status This bit provides the clock status of FIRC in partition 1. 0b - Clock is not running. 1b - Clock is running.
19 BLOCK51	IP block status This bit provides the clock status of SXOSC in partition 1. 0b - Clock is not running. 1b - Clock is running.
18	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK50	This bit provides the clock status of SIRC in partition 1. 0b - Clock is not running. 1b - Clock is running.
17 BLOCK49	IP block status This bit provides the clock status of TSPC in partition 1. 0b - Clock is not running. 1b - Clock is running.
16 —	Reserved This field is reserved and read returns zeros.
15 BLOCK47	IP block status This bit provides the clock status of CMU_0-5 in partition 1. 0b - Clock is not running. 1b - Clock is running.
14 —	Reserved This field is reserved and read returns zeros.
13 BLOCK45	IP block status This bit provides the clock status of WKPU in partition 1. 0b - Clock is not running. 1b - Clock is running.
12 —	Reserved This field is reserved and read returns zeros.
11 BLOCK43	IP block status This bit provides the clock status of DCM in partition 1. 0b - Clock is not running. 1b - Clock is running.
10 BLOCK42	IP block status This bit provides the clock status of SIUL_VIRTWRAPPER_PDAC3 in partition 1. 0b - Clock is not running. 1b - Clock is running.
9	IP block status This bit provides the clock status of SIUL_VIRTWRAPPER_PDAC2_M7_1 in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK41	0b - Clock is not running. 1b - Clock is running.
8 BLOCK40	IP block status This bit provides the clock status of SIUL_VIRTWRAPPER_PDAC2_M7_1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK39	IP block status This bit provides the clock status of SIUL_VIRTWRAPPER_PDAC1_M7_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK38	IP block status This bit provides the clock status of SIUL_VIRTWRAPPER_PDAC1_M7_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK37	IP block status This bit provides the clock status of SIUL_VIRTWRAPPER_PDAC0_HSE in partition 1. 0b - Clock is not running. 1b - Clock is running.
4 BLOCK36	IP block status This bit provides the clock status of SIUL_VIRTWRAPPER_PDAC0_HSE in partition 1. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK35	IP block status This bit provides the clock status of MC_RGM in partition 1. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK34	IP block status This bit provides the clock status of RTC in partition 1. 0b - Clock is not running. 1b - Clock is running.
1	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK33	This bit provides the clock status of DMAMUX_1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK32	IP block status This bit provides the clock status of DMAMUX_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.

43.7.27 Partition 1 COFB Set 2 Clock Status Register (PRTN1_COFB2_STAT)

Offset

Register	Offset
PRTN1_COFB2_STAT	318h

Function

This register provides the status of set 2 of COFBs inside partition 1.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK95	0	BLOCK93	BLOCK92	BLOCK91	0	BLOCK89	BLOCK88	BLOCK87	BLOCK86	BLOCK85	BLOCK84	0	0	BLOCK81	BLOCK80
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK79	BLOCK78	BLOCK77	BLOCK76	BLOCK75	BLOCK74	BLOCK73	0	0	BLOCK70	BLOCK69	BLOCK68	BLOCK67	BLOCK66	BLOCK65	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BLOCK95	IP block status This bit provides the clock status of TempSense in partition 1. 0b - Clock is not running. 1b - Clock is running.
30 —	Reserved This field is reserved and read returns zeros.
29 BLOCK93	IP block status This bit provides the clock status of LPCMP_1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
28 BLOCK92	IP block status This bit provides the clock status of LPCMP_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK91	IP block status This bit provides the clock status of SAI_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
26 —	Reserved This field is reserved and read returns zeros.
25 BLOCK89	IP block status This bit provides the clock status of LPSPI_3 in partition 1. 0b - Clock is not running. 1b - Clock is running.
24 BLOCK88	IP block status This bit provides the clock status of LPSPI_2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
23 BLOCK87	IP block status This bit provides the clock status of LPSPI_1 in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
22 BLOCK86	IP block status This bit provides the clock status of LPSPI_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
21 BLOCK85	IP block status This bit provides the clock status of LPI2C_1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
20 BLOCK84	IP block status This bit provides the clock status of LPI2C_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
19 —	Reserved This field is reserved and read returns zeros.
18 —	Reserved This field is reserved and read returns zeros.
17 BLOCK81	IP block status This bit provides the clock status of LPUART_7 in partition 1. 0b - Clock is not running. 1b - Clock is running.
16 BLOCK80	IP block status This bit provides the clock status of LPUART_6 in partition 1. 0b - Clock is not running. 1b - Clock is running.
15 BLOCK79	IP block status This bit provides the clock status of LPUART_5 in partition 1. 0b - Clock is not running. 1b - Clock is running.
14	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK78	This bit provides the clock status of LPUART_4 in partition 1. 0b - Clock is not running. 1b - Clock is running.
13 BLOCK77	IP block status This bit provides the clock status of LPUART_3 in partition 1. 0b - Clock is not running. 1b - Clock is running.
12 BLOCK76	IP block status This bit provides the clock status of LPUART_2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
11 BLOCK75	IP block status This bit provides the clock status of LPUART_1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
10 BLOCK74	IP block status This bit provides the clock status of LPUART_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK73	IP block status This bit provides the clock status of FlexIO in partition 1. 0b - Clock is not running. 1b - Clock is running.
8 —	Reserved This field is reserved and read returns zeros.
7 —	Reserved This field is reserved and read returns zeros.
6 BLOCK70	IP block status This bit provides the clock status of FlexCAN_5 in partition 1. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 BLOCK69	IP block status This bit provides the clock status of FlexCAN_4 in partition 1. 0b - Clock is not running. 1b - Clock is running.
4 BLOCK68	IP block status This bit provides the clock status of FlexCAN_3 in partition 1. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK67	IP block status This bit provides the clock status of FlexCAN_2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK66	IP block status This bit provides the clock status of FlexCAN_1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
1 BLOCK65	IP block status This bit provides the clock status of FlexCAN_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
0 —	Reserved This field is reserved and read returns zeros.

43.7.28 Partition 1 COFB Set 3 Clock Status Register (PRTN1_COFB3_STAT)

Offset

Register	Offset
PRTN1_COFB3_STAT	31Ch

Function

This register provides the status of set 3 of COFBs inside partition 1.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 1.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	BLOCK110	0	BLOCK108	BLOCK107	BLOCK106	BLOCK105	BLOCK104	BLOCK103	BLOCK102	BLOCK101	0	BLOCK99	BLOCK98	BLOCK97	BLOCK96
W																
Reset	0	1	0	1	1	1	1	1	1	1	1	0	1	1	1	0

Fields

Field	Function
31 —	Reserved This field is reserved and read returns zeros.
30 —	Reserved This field is reserved and read returns zeros.
29 —	Reserved This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 —	Reserved This field is reserved and read returns zeros.
26 —	Reserved This field is reserved and read returns zeros.
25 —	Reserved This field is reserved and read returns zeros.
24 —	Reserved This field is reserved and read returns zeros.
23 —	Reserved This field is reserved and read returns zeros.
22 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved This field is reserved and read returns zeros.
20 —	Reserved This field is reserved and read returns zeros.
19 —	Reserved This field is reserved and read returns zeros.
18 —	Reserved This field is reserved and read returns zeros.
17 —	Reserved This field is reserved and read returns zeros.
16 —	Reserved This field is reserved and read returns zeros.
15 —	Reserved This field is reserved and read returns zeros.
14 BLOCK110	IP block status This bit provides the clock status of Reserved block in partition 1. 0b - Clock is not running. 1b - Clock is running.
13 —	Reserved This field is reserved and read returns zeros.
12 BLOCK108	IP block status This bit provides the clock status of SELF-TEST GPR in partition 1. 0b - Clock is not running. 1b - Clock is running.
11 BLOCK107	IP block status This bit provides the clock status of Reserved block in partition 1. 0b - Clock is not running. 1b - Clock is running.
10 BLOCK106	IP block status This bit provides the clock status of Reserved block in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
9 BLOCK105	IP block status This bit provides the clock status of Reserved block in partition 1. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK104	IP block status This bit provides the clock status of STCU2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK103	IP block status This bit provides the clock status of Reserved block in partition 1. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK102	IP block status This bit provides the clock status of Reserved block in partition 1. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK101	IP block status This bit provides the clock status of JDC in partition 1. 0b - Clock is not running. 1b - Clock is running.
4 —	Reserved This field is reserved and read returns zeros.
3 BLOCK99	IP block status This bit provides the clock status of HSE in partition 1. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK98	IP block status This bit provides the clock status of MTR in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
1 BLOCK97	IP block status This bit provides the clock status of FCCU in partition 1. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK96	IP block status This bit provides the clock status of CRC in partition 1. 0b - Clock is not running. 1b - Clock is running.

43.7.29 Partition 1 COFB Set 0 Clock Enable Register (PRTN1_COFB0_CLKEN)

Offset

Register	Offset
PRTN1_COFB0_CLKEN	330h

Function

This register provides clock control signaling to the individual COFBs in set 0 inside partition 1. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REQ31	0	REQ29	REQ28	0	0	0	REQ24	REQ23	REQ22	REQ21	0	0	0	0	0
W																
Reset	0	1	0	1	1	1	1	0	0	0	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ15	REQ14	REQ13	REQ12	REQ11	REQ10	REQ9	REQ8	REQ7	REQ6	REQ5	REQ4	REQ3	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31 REQ31	Clock enable This bit provides the clock enable control for INTM in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
30 —	Reserved This field is reserved and read returns zeros.
29 REQ29	Clock enable This bit provides the clock enable control for STM_0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
28 REQ28	Clock enable This bit provides the clock enable control for SWT_0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
27 —	Reserved This field is reserved and read returns zeros.
26 —	Reserved This field is reserved and read returns zeros.
25 —	Reserved This field is reserved and read returns zeros.
24 REQ24	Clock enable This bit provides the clock enable control for MSCM in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
23 REQ23	Clock enable This bit provides the clock enable control for ERM in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
22 REQ22	Clock enable This bit provides the clock enable control for EIM in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
21 REQ21	Clock enable This bit provides the clock enable control for SDA-AP in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
20 —	Reserved This field is reserved and read returns zeros.
19 —	Reserved This field is reserved and read returns zeros.
18 —	Reserved This field is reserved and read returns zeros.
17 —	Reserved This field is reserved and read returns zeros.
16 —	Reserved This field is reserved and read returns zeros.
15 REQ15	Clock enable This bit provides the clock enable control for eDMA in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
14 REQ14	Clock enable This bit provides the clock enable control for eDMA in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
13 REQ13	Clock enable This bit provides the clock enable control for eDMA in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
12 REQ12	Clock enable This bit provides the clock enable control for eDMA in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
11 REQ11	Clock enable This bit provides the clock enable control for eDMA in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
10 REQ10	Clock enable This bit provides the clock enable control for eDMA in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ9	Clock enable This bit provides the clock enable control for eDMA in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
8 REQ8	Clock enable This bit provides the clock enable control for eDMA in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
7 REQ7	Clock enable This bit provides the clock enable control for eDMA in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
6 REQ6	Clock enable This bit provides the clock enable control for eDMA in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
5 REQ5	Clock enable This bit provides the clock enable control for eDMA in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
4	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ4	This bit provides the clock enable control for eDMA in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
3 REQ3	Clock enable This bit provides the clock enable control for eDMA in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 —	Reserved This field is reserved and read returns zeros.

43.7.30 Partition 1 COFB Set 1 Clock Enable Register (PRTN1_COFB1_CLKEN)

Offset

Register	Offset
PRTN1_COFB1_CLKEN	334h

Function

This register provides clock control signaling to the individual COFBs in set 1 inside partition 1. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REQ63	0	0	0	0	0	0	REQ56	0	0	REQ53	0	REQ51	0	REQ49	0
W																
Reset	0	0	0	1	1	1	0	0	1	1	1	1	1	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ47	0	REQ45	0	0	REQ42	0	0	0	0	0	0	0	REQ34	REQ33	REQ32
W																
Reset	0	0	1	0	1	1	1	1	1	1	1	1	1	1	0	0

Fields

Field	Function
31 REQ63	Clock enable This bit provides the clock enable control for PIT_2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
30 —	Reserved This field is reserved and read returns zeros.
29 —	Reserved This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 —	Reserved This field is reserved and read returns zeros.
26 —	Reserved This field is reserved and read returns zeros.
25 —	Reserved This field is reserved and read returns zeros.
24 REQ56	Clock enable This bit provides the clock enable control for PLL in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
23	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
22 —	Reserved This field is reserved and read returns zeros.
21 REQ53	Clock enable This bit provides the clock enable control for FXOSC in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
20 —	Reserved This field is reserved and read returns zeros.
19 REQ51	Clock enable This bit provides the clock enable control for SXOSC in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
18 —	Reserved This field is reserved and read returns zeros.
17 REQ49	Clock enable This bit provides the clock enable control for TSPC in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
16 —	Reserved This field is reserved and read returns zeros.
15 REQ47	Clock enable This bit provides the clock enable control for CMU_0-5 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
14 —	Reserved This field is reserved and read returns zeros.
13 REQ45	Clock enable This bit provides the clock enable control for WKPU in partition 1. 0b - Clock is turned off.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock is turned on.
12 —	Reserved This field is reserved and read returns zeros.
11 —	Reserved This field is reserved and read returns zeros.
10 REQ42	Clock enable This bit provides the clock enable control for SIUL_VIRTWRAPPER_PDAC3 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
9 —	Reserved This field is reserved and read returns zeros.
8 —	Reserved This field is reserved and read returns zeros.
7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 REQ34	Clock enable This bit provides the clock enable control for RTC in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
1 REQ33	Clock enable This bit provides the clock enable control for DMAMUX_1 in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
0 REQ32	Clock enable This bit provides the clock enable control for DMAMUX_0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.

43.7.31 Partition 1 COFB Set 2 Clock Enable Register (PRTN1_COFB2_CLKEN)

Offset

Register	Offset
PRTN1_COFB2_CLKEN	338h

Function

This register provides clock control signaling to the individual COFBs in set 2 inside partition 1. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REQ95	0	REQ93	REQ92	REQ91	0	REQ89	REQ88	REQ87	REQ86	REQ85	REQ84	0	0	REQ81	REQ80
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ79	REQ78	REQ77	REQ76	REQ75	REQ74	REQ73	0	0	REQ70	REQ69	REQ68	REQ67	REQ66	REQ65	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 REQ95	Clock enable This bit provides the clock enable control for TempSense in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
30 —	Reserved This field is reserved and read returns zeros.
29 REQ93	Clock enable This bit provides the clock enable control for LPCMP_1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
28 REQ92	Clock enable This bit provides the clock enable control for LPCMP_0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
27 REQ91	Clock enable This bit provides the clock enable control for SAI_0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
26 —	Reserved This field is reserved and read returns zeros.
25 REQ89	Clock enable This bit provides the clock enable control for LPSPI_3 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
24 REQ88	Clock enable This bit provides the clock enable control for LPSPI_2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
23 REQ87	Clock enable This bit provides the clock enable control for LPSPI_1 in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
22 REQ86	Clock enable This bit provides the clock enable control for LPSPI_0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
21 REQ85	Clock enable This bit provides the clock enable control for LPI2C_1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
20 REQ84	Clock enable This bit provides the clock enable control for LPI2C_0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
19 —	Reserved This field is reserved and read returns zeros.
18 —	Reserved This field is reserved and read returns zeros.
17 REQ81	Clock enable This bit provides the clock enable control for LPUART_7 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
16 REQ80	Clock enable This bit provides the clock enable control for LPUART_6 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
15 REQ79	Clock enable This bit provides the clock enable control for LPUART_5 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
14	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ78	This bit provides the clock enable control for LPUART_4 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
13 REQ77	Clock enable This bit provides the clock enable control for LPUART_3 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
12 REQ76	Clock enable This bit provides the clock enable control for LPUART_2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
11 REQ75	Clock enable This bit provides the clock enable control for LPUART_1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
10 REQ74	Clock enable This bit provides the clock enable control for LPUART_0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ73	Clock enable This bit provides the clock enable control for FlexIO in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
8 —	Reserved This field is reserved and read returns zeros.
7 —	Reserved This field is reserved and read returns zeros.
6 REQ70	Clock enable This bit provides the clock enable control for FlexCAN_5 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 REQ69	Clock enable This bit provides the clock enable control for FlexCAN_4 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
4 REQ68	Clock enable This bit provides the clock enable control for FlexCAN_3 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
3 REQ67	Clock enable This bit provides the clock enable control for FlexCAN_2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
2 REQ66	Clock enable This bit provides the clock enable control for FlexCAN_1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
1 REQ65	Clock enable This bit provides the clock enable control for FlexCAN_0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
0 —	Reserved This field is reserved and read returns zeros.

43.7.32 Partition 1 COFB Set 3 Clock Enable Register (PRTN1_COFB3_CLKEN)

Offset

Register	Offset
PRTN1_COFB3_CLKEN	33Ch

Function

This register provides clock control signaling to the individual COFBs in set 3 inside partition 1. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	REQ104	0	REQ102	0	0	0	0	0	REQ96
W																
Reset	0	1	0	1	1	1	1	1	1	1	1	0	1	1	1	0

Fields

Field	Function
31 —	Reserved This field is reserved and read returns zeros.
30 —	Reserved This field is reserved and read returns zeros.
29 —	Reserved This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 —	Reserved This field is reserved and read returns zeros.
26 —	Reserved This field is reserved and read returns zeros.
25 —	Reserved This field is reserved and read returns zeros.
24 —	Reserved This field is reserved and read returns zeros.
23	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
22 —	Reserved This field is reserved and read returns zeros.
21 —	Reserved This field is reserved and read returns zeros.
20 —	Reserved This field is reserved and read returns zeros.
19 —	Reserved This field is reserved and read returns zeros.
18 —	Reserved This field is reserved and read returns zeros.
17 —	Reserved This field is reserved and read returns zeros.
16 —	Reserved This field is reserved and read returns zeros.
15 —	Reserved This field is reserved and read returns zeros.
14 —	Reserved This field is reserved and read returns zeros.
13 —	Reserved This field is reserved and read returns zeros.
12 —	Reserved This field is reserved and read returns zeros.
11 —	Reserved This field is reserved and read returns zeros.
10 —	Reserved This field is reserved and read returns zeros.
9 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 REQ104	Clock enable This bit provides the clock enable control for STCU2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
7 —	Reserved This field is reserved and read returns zeros.
6 REQ102	Clock enable This bit provides the clock enable control for Reserved block in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 REQ96	Clock enable This bit provides the clock enable control for CRC in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.

43.7.33 Partition 2 Process Configuration Register (PRTN2_PCONF)

Offset

Register	Offset
PRTN2_PCONF	500h

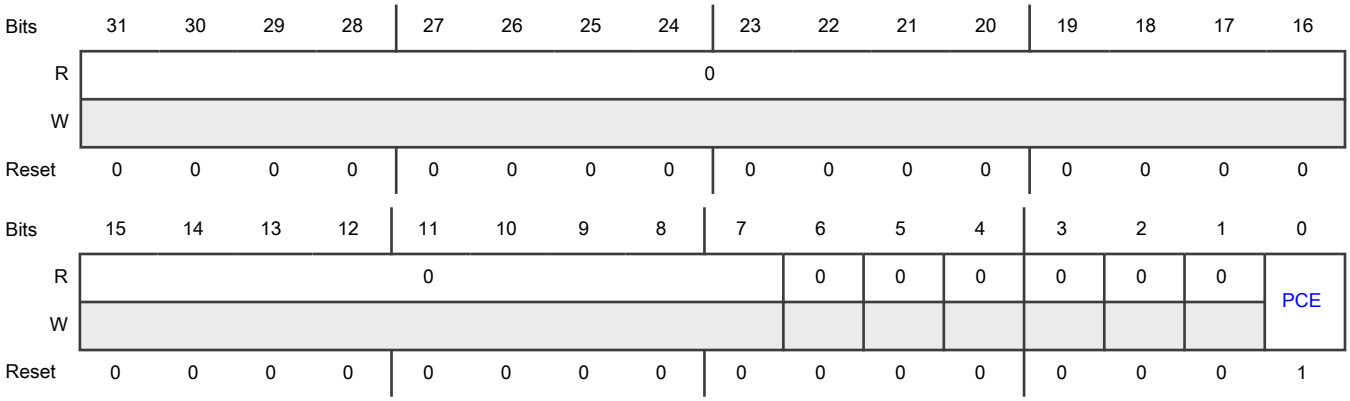
Function

This register provides a configuration for the hardware processes corresponding to partition 2. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN2_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN2_PCONF and PRTN2_PUPD registers are used to determine the hardware processes to be executed. These are triggered in parallel and independent of each other. All dependent processes should be requested one after another from the software.

NOTE

The partition clock enable/disable and output safe stating enable/disable are not standalone and must be done coherently in a fixed sequence. For details, see Software Reset Partition Turn-On Flow Chart and Software reset partition turn-off flowchart in Reset chapter.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 —	Reserved This field is reserved and read returns zeros.
0 PCE	Partition clock enable This bit controls whether the clock to IPs (other than core(s)) in the partition should be enabled or disabled. 0b - Disable the clock to IPs 1b - Enable the clock to IPs

43.7.34 Partition 2 Process Update Register (PRTN2_PUPD)

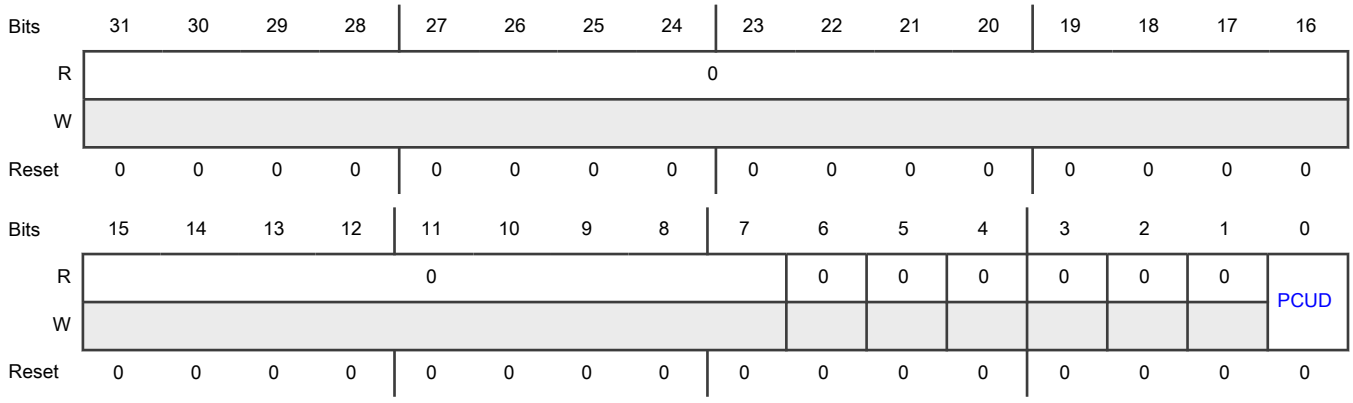
Offset

Register	Offset
PRTN2_PUPD	504h

Function

This register provides trigger signaling for the hardware processes corresponding to partition 2. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN2_PCONF register. When the hardware process is finished the corresponding bit in this register is auto-cleared to logic-0.

Diagram



Fields

Field	Function
31-7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCUD	Partition clock update This bit controls whether the hardware processes for enabling/disabling the clock to IPs (other than core(s)) in the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

43.7.35 Partition 2 Status Register (PRTN2_STAT)

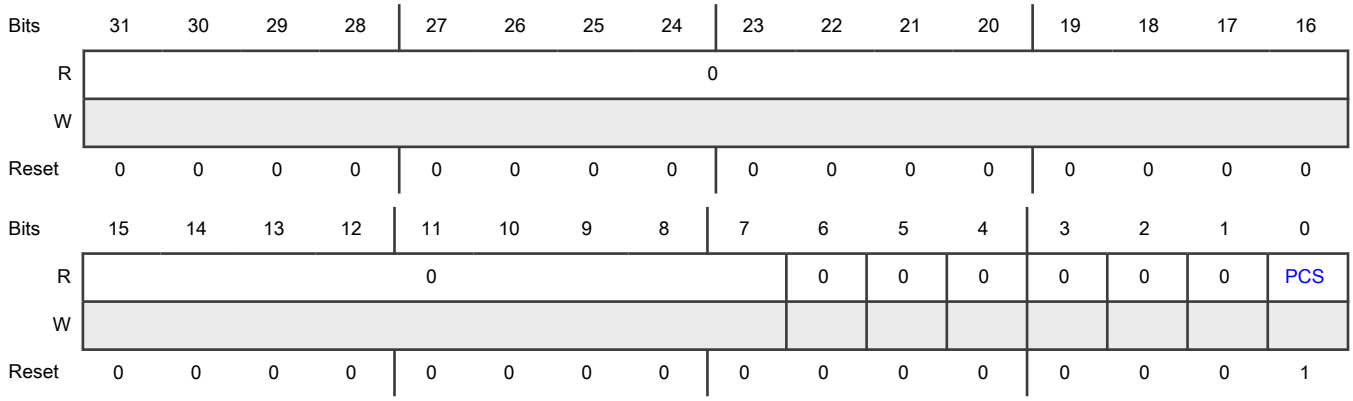
Offset

Register	Offset
PRTN2_STAT	508h

Function

This register provides the current status of the control signals from the partition 2.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCS	Partition clock status This bit provides the status of the clock to partition. 0b - Clock is inactive 1b - Clock is active

43.7.36 Partition 2 COFB Set 0 Clock Status Register (PRTN2_COFB0_STAT)

Offset

Register	Offset
PRTN2_COFB0_STAT	510h

Function

This register provides the status of set 0 of COFBs inside partition 2.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 2.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	BLOCK29	0	BLOCK27	0	BLOCK25	BLOCK24	BLOCK23	BLOCK22	BLOCK21	BLOCK20	BLOCK19	BLOCK18	BLOCK17	BLOCK16
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK15	BLOCK14	BLOCK13	BLOCK12	BLOCK11	BLOCK10	BLOCK9	BLOCK8	BLOCK7	BLOCK6	BLOCK5	BLOCK4	0	0	BLOCK1	BLOCK0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31 —	Reserved This field is reserved and read returns zeros.
30 —	Reserved This field is reserved and read returns zeros.
29 BLOCK29	IP block status This bit provides the clock status of STM_1 in partition 2. 0b - Clock is not running. 1b - Clock is running.
28 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 BLOCK27	IP block status This bit provides the clock status of SWT_1 in partition 2. 0b - Clock is not running. 1b - Clock is running.
26 —	Reserved This field is reserved and read returns zeros.
25 BLOCK25	IP block status This bit provides the clock status of PRAM_1 in partition 2. 0b - Clock is not running. 1b - Clock is running.
24 BLOCK24	IP block status This bit provides the clock status of SEMA42 in partition 2. 0b - Clock is not running. 1b - Clock is running.
23 BLOCK23	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
22 BLOCK22	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
21 BLOCK21	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
20 BLOCK20	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
19	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK19	This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
18 BLOCK18	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
17 BLOCK17	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
16 BLOCK16	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
15 BLOCK15	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
14 BLOCK14	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
13 BLOCK13	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
12 BLOCK12	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 BLOCK11	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
10 BLOCK10	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK9	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK8	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK7	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK6	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK5	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running. 1b - Clock is running.
4 BLOCK4	IP block status This bit provides the clock status of eDMA in partition 2. 0b - Clock is not running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock is running.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 BLOCK1	IP block status This bit provides the clock status of XBIC in partition 2. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK0	IP block status This bit provides the clock status of XBIC in partition 2. 0b - Clock is not running. 1b - Clock is running.

43.7.37 Partition 2 COFB Set 1 Clock Status Register (PRTN2_COFB1_STAT)

Offset

Register	Offset
PRTN2_COFB1_STAT	514h

Function

This register provides the status of set 1 of COFBs inside partition 2.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 2.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK63	BLOCK62	0	0	BLOCK59	BLOCK58	0	0	BLOCK55	0	0	0	BLOCK51	0	0	BLOCK48
W																
Reset	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK47	0	0	0	0	BLOCK42	BLOCK41	BLOCK40	BLOCK39	BLOCK38	BLOCK37	BLOCK36	BLOCK35	0	0	BLOCK32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BLOCK63	IP block status This bit provides the clock status of CM7_1_TCM in partition 2. 0b - Clock is not running. 1b - Clock is running.
30 BLOCK62	IP block status This bit provides the clock status of CM7_0_TCM in partition 2. 0b - Clock is not running. 1b - Clock is running.
29 —	Reserved This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 BLOCK59	IP block status This bit provides the clock status of HSE in partition 2. 0b - Clock is not running. 1b - Clock is running.
26 BLOCK58	IP block status This bit provides the clock status of LPCMP_2 in partition 2. 0b - Clock is not running. 1b - Clock is running.
25	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
24 —	Reserved This field is reserved and read returns zeros.
23 BLOCK55	IP block status This bit provides the clock status of SAI_1 in partition 2. 0b - Clock is not running. 1b - Clock is running.
22 —	Reserved This field is reserved and read returns zeros.
21 —	Reserved This field is reserved and read returns zeros.
20 —	Reserved This field is reserved and read returns zeros.
19 BLOCK51	IP block status This bit provides the clock status of QuadSPI in partition 2. 0b - Clock is not running. 1b - Clock is running.
18 —	Reserved This field is reserved and read returns zeros.
17 —	Reserved This field is reserved and read returns zeros.
16 BLOCK48	IP block status This bit provides the clock status of LPSPI_5 in partition 2. 0b - Clock is not running. 1b - Clock is running.
15 BLOCK47	IP block status This bit provides the clock status of LPSPI_4 in partition 2. 0b - Clock is not running. 1b - Clock is running.
14	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
13 —	Reserved This field is reserved and read returns zeros.
12 —	Reserved This field is reserved and read returns zeros.
11 —	Reserved This field is reserved and read returns zeros.
10 BLOCK42	IP block status This bit provides the clock status of LPUART_15 in partition 2. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK41	IP block status This bit provides the clock status of LPUART_14 in partition 2. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK40	IP block status This bit provides the clock status of LPUART_13 in partition 2. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK39	IP block status This bit provides the clock status of LPUART_12 in partition 2. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK38	IP block status This bit provides the clock status of LPUART_11 in partition 2. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK37	IP block status This bit provides the clock status of LPUART_10 in partition 2. 0b - Clock is not running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock is running.
4 BLOCK36	IP block status This bit provides the clock status of LPUART_9 in partition 2. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK35	IP block status This bit provides the clock status of LPUART_8 in partition 2. 0b - Clock is not running. 1b - Clock is running.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 BLOCK32	IP block status This bit provides the clock status of EMAC in partition 2. 0b - Clock is not running. 1b - Clock is running.

43.7.38 Partition 2 COFB Set 0 Clock Enable Register (PRTN2_COFB0_CLKEN)

Offset

Register	Offset
PRTN2_COFB0_CLKEN	530h

Function

This register provides clock control signaling to the individual COFBs in set 0 inside partition 2. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	REQ29	0	REQ27	0	0	REQ24	REQ23	REQ22	REQ21	REQ20	REQ19	REQ18	REQ17	REQ16
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ15	REQ14	REQ13	REQ12	REQ11	REQ10	REQ9	REQ8	REQ7	REQ6	REQ5	REQ4	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31 —	Reserved This field is reserved and read returns zeros.
30 —	Reserved This field is reserved and read returns zeros.
29 REQ29	Clock enable This bit provides the clock enable control for STM_1 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
28 —	Reserved This field is reserved and read returns zeros.
27 REQ27	Clock enable This bit provides the clock enable control for SWT_1 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
26 —	Reserved This field is reserved and read returns zeros.
25 —	Reserved This field is reserved and read returns zeros.
24 REQ24	Clock enable This bit provides the clock enable control for SEMA42 in partition 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
23 REQ23	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
22 REQ22	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
21 REQ21	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
20 REQ20	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
19 REQ19	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
18 REQ18	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
17 REQ17	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
16	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ16	This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
15 REQ15	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
14 REQ14	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
13 REQ13	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
12 REQ12	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
11 REQ11	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
10 REQ10	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ9	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 REQ8	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
7 REQ7	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
6 REQ6	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
5 REQ5	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
4 REQ4	Clock enable This bit provides the clock enable control for eDMA in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 —	Reserved This field is reserved and read returns zeros.

43.7.39 Partition 2 COFB Set 1 Clock Enable Register (PRTN2_COFB1_CLKEN)

Offset

Register	Offset
PRTN2_COFB1_CLKEN	534h

Function

This register provides clock control signaling to the individual COFBs in set 1 inside partition 2. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REQ63	REQ62	0	0	0	REQ58	0	0	REQ55	0	0	0	REQ51	0	0	REQ48
W																
Reset	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ47	0	0	0	0	REQ42	REQ41	REQ40	REQ39	REQ38	REQ37	REQ36	REQ35	0	0	REQ32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 REQ63	Clock enable This bit provides the clock enable control for CM7_1_TCM in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
30 REQ62	Clock enable This bit provides the clock enable control for CM7_0_TCM in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
29	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 —	Reserved This field is reserved and read returns zeros.
26 REQ58	Clock enable This bit provides the clock enable control for LPCMP_2 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
25 —	Reserved This field is reserved and read returns zeros.
24 —	Reserved This field is reserved and read returns zeros.
23 REQ55	Clock enable This bit provides the clock enable control for SAI_1 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
22 —	Reserved This field is reserved and read returns zeros.
21 —	Reserved This field is reserved and read returns zeros.
20 —	Reserved This field is reserved and read returns zeros.
19 REQ51	Clock enable This bit provides the clock enable control for QuadSPI in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
18 —	Reserved This field is reserved and read returns zeros.
17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
16 REQ48	Clock enable This bit provides the clock enable control for LPSPI_5 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
15 REQ47	Clock enable This bit provides the clock enable control for LPSPI_4 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
14 —	Reserved This field is reserved and read returns zeros.
13 —	Reserved This field is reserved and read returns zeros.
12 —	Reserved This field is reserved and read returns zeros.
11 —	Reserved This field is reserved and read returns zeros.
10 REQ42	Clock enable This bit provides the clock enable control for LPUART_15 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ41	Clock enable This bit provides the clock enable control for LPUART_14 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
8 REQ40	Clock enable This bit provides the clock enable control for LPUART_13 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
7	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ39	This bit provides the clock enable control for LPUART_12 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
6 REQ38	Clock enable This bit provides the clock enable control for LPUART_11 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
5 REQ37	Clock enable This bit provides the clock enable control for LPUART_10 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
4 REQ36	Clock enable This bit provides the clock enable control for LPUART_9 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
3 REQ35	Clock enable This bit provides the clock enable control for LPUART_8 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 REQ32	Clock enable This bit provides the clock enable control for EMAC in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.

43.8 Glossary

WFI Wait for interrupt

COFB Collection of functional blocks also referred as number of peripherals

Chapter 44
Power Control Unit (MC_PCU)

44.1 Introduction

The power control unit (MC_PCU) is used for initiating a Standby mode entry that reduces the overall chip power consumption. Power can be saved by disconnecting parts of the chip from the power supply. The blocks inside the chip are grouped into multiple parts having this capability, which are called "power domains".

When a power domain is disconnected from the supply, the power consumption is reduced to zero in that domain. Any status information, such as, power domain is lost. When you reconnect a power domain to the supply voltage, the domain draws an increased current until the power domain reaches its operational voltage. Maximum power saving is achieved by entering the Standby mode.

After the MC_ME asserts a standby entry request, MC_PCU initiates the power sequence, which is non-retractable and includes the handshake with the chip power management controller. The power-up/down sequences are handled by FSMs to ensure a smooth and safe transition into and out of the Standby mode. Exiting the Standby mode can only be done through a system wakeup event, power-on reset, destructive reset, or a functional reset.

44.2 Power sequence FSM

MC_PCU implements an FSM to initiate the power sequencing of the Standby mode entry/exit sequence for the chip.

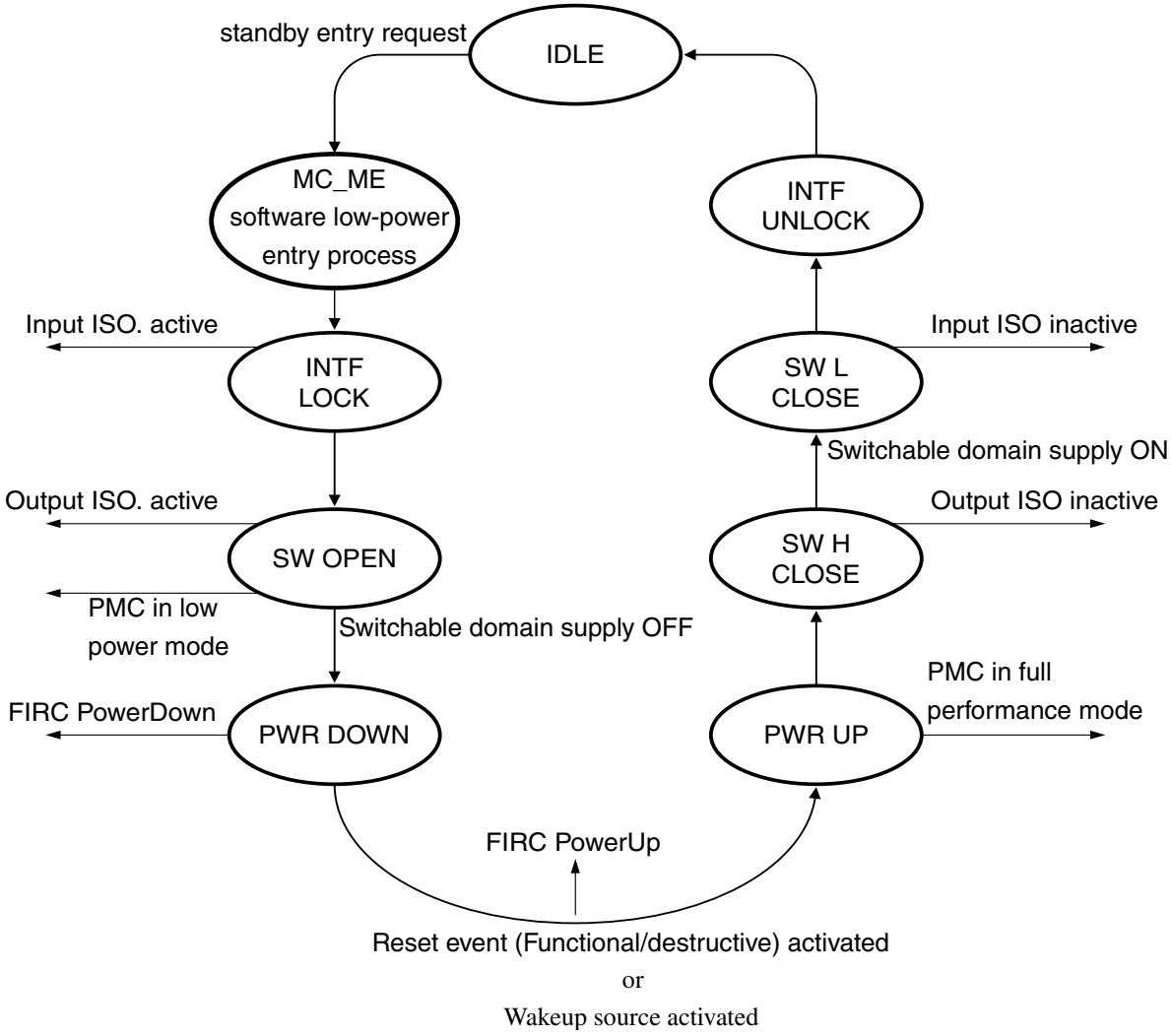


Figure 173. MC_PCU FSM

NOTE

When destructive reset is asserted from MC_RGM, MC_PCU FSM moves to the IDLE state immediately. Indication of this is not shown in the above figure.

Table 217. MC_PCU FSM transition description

State	Name	Exit condition	Signal controlled	Signal monitored
IDLE	Idle state	Standby request received from MC_ME	-	-
MC_ME software low power entry process	MC_ME software low power entry process	Software entry sequence completed (SW4 process completed)	-	-

Table continues on the next page...

Table 217. MC_PCU FSM transition description (continued)

State	Name	Exit condition	Signal controlled	Signal monitored
INTF LOCK	Interface lock state	Input isolation active	Input isolation activation; Switchable power-domain reset	Input isolation active
SW OPEN	Switch open state	Output isolation active, Switchable domain supply turned off	Output isolation activation, low power mode request to PMC	Output isolation active, Switchable domain supply
PWR DOWN	Power down state	Wakeup/functional reset occurrence	SIRC power down	Wakeup and functional reset
PWR UP	Power up state	PMC in full performance mode	-	-
SW H CLOSE	Switch H close state	Switchable domain supply turned on	-	Switchable domain supply
SW L CLOSE	Switch L close state	Output isolation inactive	Output isolation activation	Output isolation active
INTF UNLOCK	Interface unlock	Input isolation inactive	Input isolation activation.	Input isolation active

44.3 Glossary

FSM Finite state machine

Chapter 45
Wakeup Unit (WKPU)

45.1 Chip-specific WKPU information

45.1.1 WKPU configuration on the chip

WKPU provides the following features:

- A configurable, low-power wake-up capability to the chip from multiple configurable asynchronous wake-up events.
- Support for four internal and minimum 59 external source depending on the chip, that can generate interrupts or wake-up events.
- Support for an NMI input.

Table 218. WKPU configuration on the chip

Number of sources, interrupts, and vectors	MWCT2D17S/ MWCT2D16S/ MWCT2016S/ MWCT2015S/ MWCT2014S	Description
NMI sources	1	Single NMI pin routed to all application cores
Internal wake-up sources	4	• SWT_0 wake-up, RTC-API wake-up • RTC timeout • Analog comparator round robin wake-up (from LPCMP_0, 1, and 2 round robin wake-up)[1] • RTI wake-up
External wake-up sources	60	Minimum 59 GPIO pins (WKPU[0]-WKPU[59])[2] support the wake-up functionality. See the IOMUX file attached to this document for details.
Glitch filters for external interrupts	60	Glitch filter on each external wake-up source
External interrupt vectors	8	—

1. Both wake-up sources 0 (RTC-API) and 2 (CMP_*x* trigger mode wake-up) use RTC-API wake-up, with Trigger mode having a higher priority. This means, if you configure LPCMP_*x*.RRCR0[ERR_EN] for either of the comparators, the RTC_API wake-up is used only for the CMP_*x* trigger mode operation. The RTC-API wake-up does not cause wake-up from Standby mode in this scenario.
2. For MWCT2016S, WKPU[34] is reserved.

All of the aforementioned wake-up sources can be enabled or disabled. Also, you can configure these wake-up sources, by using WKPU configuration registers, to provide wake-up on rising or falling events. See the WKPU register memory map for details.

NOTE

You must use SIUL2 to perform WKPU pin configurations (PUE, PUS, and IBE). For this, you must first configure SIUL2.MSCR[IBE] for the corresponding pin. This chip does not support WKPU-provided pin configurations; it supports only bypass control.

45.1.2 WKPU register fields and their applicability

Table 219. WKPU register fields and applicability

Register	Field	Chips where applicable
NSR	NIF1	MWCT2D17S, MWCT2D16S
NSR	NOVF1	MWCT2D17S, MWCT2D16S
NCR	NLOCK1	MWCT2D17S, MWCT2D17S
NCR	NDSS1	MWCT2D17S, MWCT2D17S
NCR	NWRE1	MWCT2D17S, MWCT2D17S
NCR	NREE1	MWCT2D17S, MWCT2D17S
NCR	NFEE1	MWCT2D17S, MWCT2D17S
NCR	NFE1	MWCT2D17S, MWCT2D17S

45.1.3 WKPU NMI configuration

WKPU supports one external source that can cause non-maskable interrupts to on-chip cores and wake-up events to the system.

The following figure shows applicable cores, some family chips may not have cores Cortex-M7_1 or Cortex-M7_2. In case of a wake-up event (internal or external), WKPU initiates the recovery of the chip and feeds an interrupt to the core(s) depending on the configuration. The sections that follow provide details related to the associated configurations.

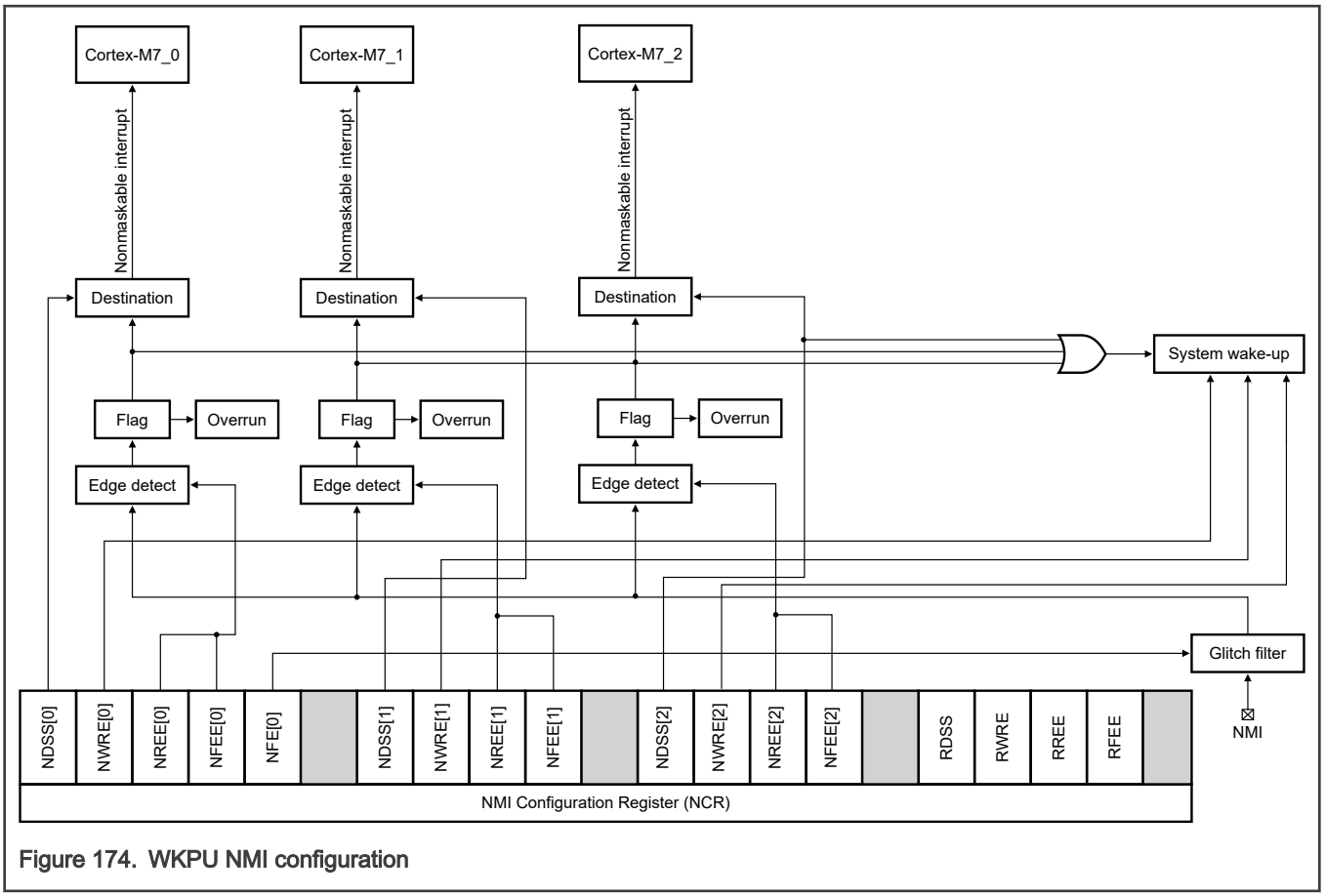


Figure 174. WKPU NMI configuration

45.1.4 WKPU wake-up source connectivity

WKPU allows the external NMI pin to assert the core NMIs on the chip. NMI supports NSR's status and overrun flags. This is what you could do using NCR:

- Control the NMI destination interrupt by configuring NCR[NDSS].
- Control the rising edge, falling edge, or either of the edge reactions to the NMI pin by using bits 2:0 of NCR[NFEE] and NCR[NREE]. The enabling of these edge reactions to the NMI pin is independent of each core.

WKPU supports the capturing of a second event per NMI input before the interrupt is cleared, thus reducing the chance of losing an NMI event.

The following figure shows routing of external wake-up events or interrupts with WKPU and the system interrupt controller.

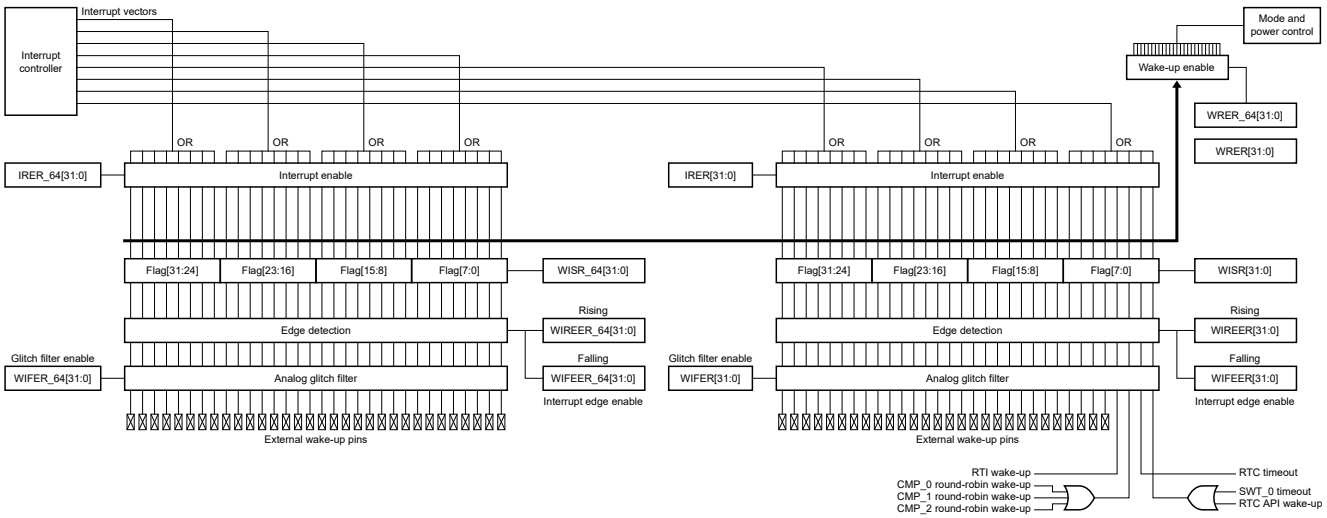


Figure 175. WKPU wake-up source connectivity

This is the wake-up source mapping to WKPU:

- Wake-up source 0 : SWT_0 timeout, RTC-API API wake-up
- Wake-up source 1 : RTC-API RTC timeout
- Wake-up source 2 : Round robin wake-up interrupt (Trigger mode interrupt) from LPCMP_0, LPCMP_1, or LPCMP_2
- Wake-up source 3 : RTI wake-up
- Wake-up source 4 : Wake-up source minimum 59-external pin wake-up sources, WKPU[0]-WKPU[59]

If you configure any or all of the LPCMP_*x*.RRCR0[RR_EN] fields to be active, the corresponding CMP_*x* pins must be dedicated for the CMP_*x* operation. In case you are not using any of the CMPs, you can use SIUL2.MSCR*x*[SSS] to configure the pins for digital functionalities.

NOTE

You must enable WKPU (by using MC_ME.PRTN*x*_COFB*y*_CLKEN) before entering any of the chip's low-power modes.

45.2 Introduction

The WKPU supports 64 external sources that can generate interrupts or wakeup events and 2 external source(s) that can cause non-maskable interrupt request(s) or wakeup event(s). In addition, it combines its wakeup events with those generated from other wakeup sources to supply a single wakeup to the system. The block diagram of WKPU and its interfaces to other system components is shown below.

NOTE

The signal widths in the following diagram might not depict this device's particular configuration. See chip-specific WKPU for more information.

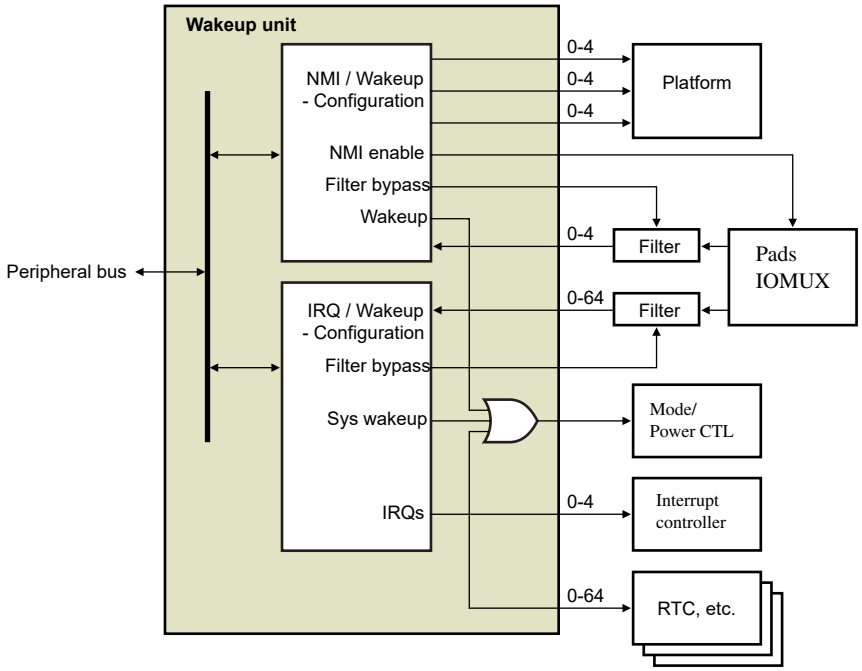


Figure 176. WKPU and connected system components

45.3 Features

The WKPU supports these features:

- Non-maskable interrupt support with:
  - 2 NMI sources
  - 2 analog glitch filters
  - Independent interrupt destination for each core:
    - Non-maskable interrupt
    - Critical interrupt
  - Active (rise/fall) edge selection control for events
  - Configurable system wakeup triggering from NMI sources
- External wakeup/interrupt support with:
  - One System interrupt vector for interrupt sources
  - 64 analog glitch filters
  - Independent interrupt mask
  - Edge detection
  - Configurable system wakeup triggering from all interrupt sources

45.4 Functional description

This section provides functional description of WKPU.

45.4.1 Non-maskable interrupts

The WKPU supports the capturing of a second event per NMI input before the interrupt is cleared, thus reducing the chance of losing an NMI event, although it creates an overrun condition.

Each NMI passes through a bypassable analog glitch filter.

NOTE

Glitch filter control and pad configuration should be performed when NMI is disabled, to avoid erroneous triggering by glitches caused by the configuration process itself.

NOTE

The figure below represents a generic configuration and might not represent this particular device's configuration. See the chip-specific information for details on this chip's WKPU.

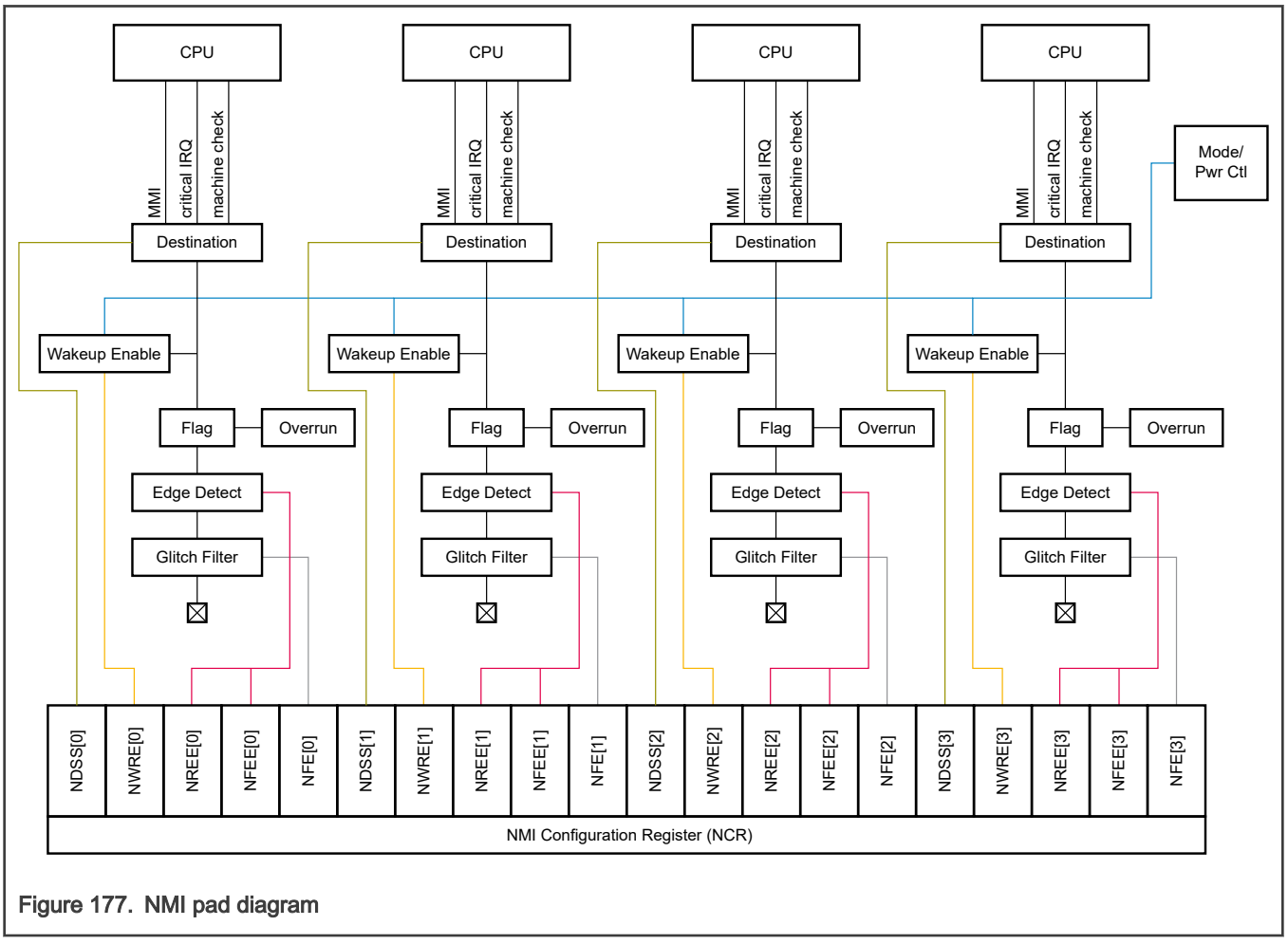


Figure 177. NMI pad diagram

45.4.1.1 NMI management

Each NMI can be enabled or disabled independently. This can be performed using the single NCR register laid out to contain all configuration bits for a given NMI in a single byte (see NMI Configuration Register (NCR)). A pad defined as an NMI can be configured by the user to recognize interrupts with an active rising edge, an active falling edge, or both edges being active. A setting of having both edge events disabled results in no interrupt being detected and should not be configured.

The active NMI edge is controlled by the user through the configuration of the NCR[NREEn] and NCR[NFEEn] bits.

NOTE

After reset, NREE and NFEE are set to '0'; therefore, the NMI functionality is disabled after reset and must be enabled explicitly by software.

After a pad's NMI functionality is enabled, the pad cannot be reconfigured to override or disable the NMI. See the chip-specific WKPU information for details of NMI implementation.

The NMI destination interrupt is controlled by the user through the configuration of the NCR[NDSSn] bits. See NMI Configuration Register (NCR) for details.

Each NMI supports a status flag and an overrun flag, which are located in the NSR register (see NMI Status Flag Register (NSR)). This register is a clear-by-write-1 register type, preventing inadvertent overwriting of other flags in the same register. The status flag is set whenever an NMI event is detected. The overrun flag is set whenever an NMI event is detected and the status flag is set (that is, status flag has not been cleared).

NOTE

The overrun flag is cleared by writing a '1' to the appropriate overrun bit in the NSR register. If the status bit is cleared and the overrun bit is still set, the pending interrupt is not cleared.

During an NMI ISR, on wakeup of the chip from an NMI, any writes to the ECC-protected memory must have the correct ECC.

45.5 External wakeups/interrupts

The WKPU supports 2 interrupt vectors to the interrupt controller of the chip. Each interrupt vector can support up to the number of external interrupt sources from the device pads, with the total across all vectors being equal to the number of external interrupt sources. Each external interrupt source is assigned to exactly one interrupt vector. The interrupt vector assignment is sequential so that one interrupt vector is for external interrupt sources 0 through N-1, the next is for N through N+M-1, and so forth.

See the following figure for an overview of the external interrupt implementation, showing an example of four interrupt vectors with eight external interrupt sources each.

NOTE

The figure below shows a generic representation of WKPU. See the chip-specific WKPU information for details on how this device's configuration might differ from this figure.

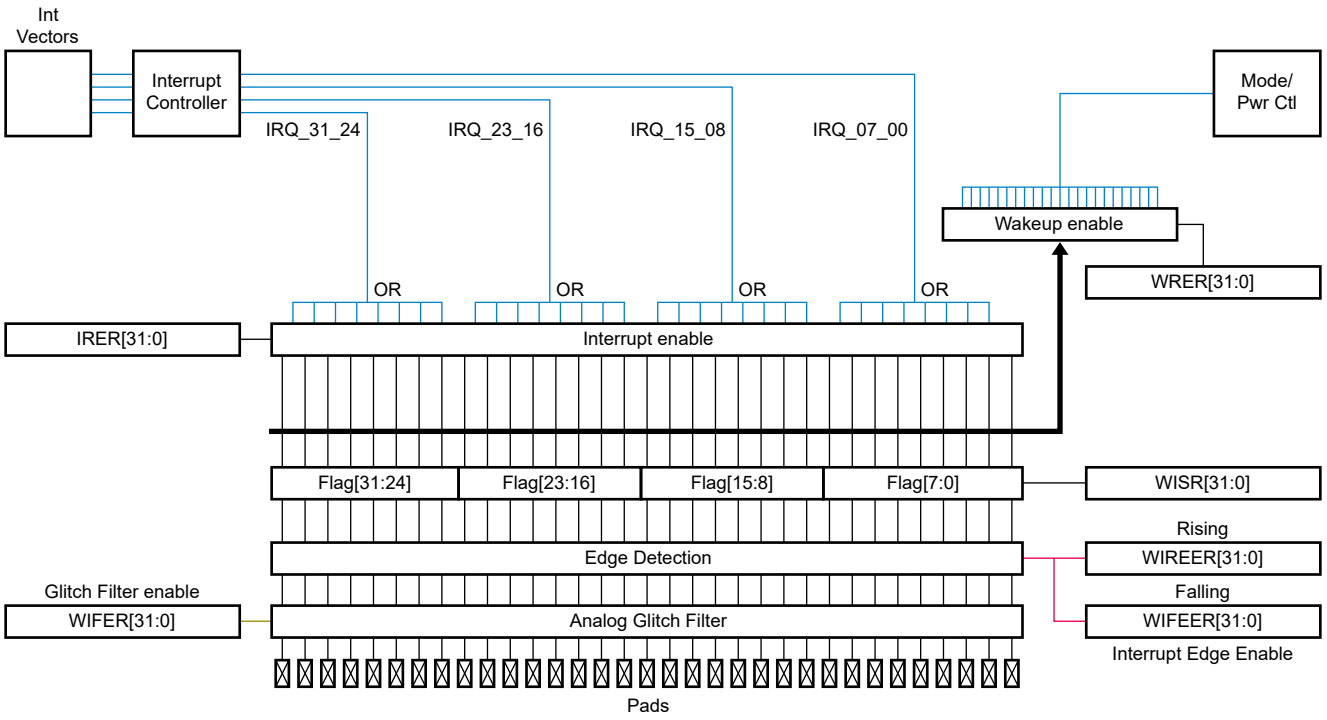


Figure 178. External interrupt pad diagram

All of the external interrupt pads within a single group have equal priority. It is the responsibility of the user software to search through the group of sources in the most appropriate way for their application.

The priority of the vectors used by the external interrupt pads is set based on the platform and the interrupt controller's priority levels, but the allocation of pads to each group of interrupts can be independently configured by the chip.

External interrupt lines have a digital glitch filter applied to them.

45.5.1 External interrupt management

Each interrupt can be enabled or disabled independently. This can be performed using a single rolled-up register (Interrupt Request Enable Register (IRER)). A pad defined as an external interrupt can be configured by the user to recognize interrupts with an active rising edge, an active falling edge, or both edges being active.

NOTE

Writing a '0' to both IREE[x] and IFEE[x] disables the external interrupt functionality for that pad completely (that is, no system wakeup or interrupt is generated on any activity of that pad).

The active IRQ edge is controlled by users through the configuration of the registers WIREER and WIFEER.

Each external interrupt supports an individual flag, which is held in the flag register, WISR. This register is a clear-by-write-1 register type, preventing inadvertent overwriting of other flags in the same register.

45.6 Initialization information

This section discusses initialization for the following features:

- Glitch filter and pad configuration
- Non-maskable interrupts

45.6.1 Glitch filter and pad configuration

Glitch filter control and pad configuration should be performed when the NMI is disabled. This is to avoid erroneous triggering by glitches caused by the configuration process.

When enabling the glitch filter, do not enable the rising-/falling-edge events bits, i.e., the NCR[NREE], NCR[NFEE], NCR[RREE], and NCR[RFEE] bits in the same register write.

45.6.2 Non-maskable interrupts

If the IBE of NMI is controlled by the NFEE or NREE fields of NMI Configuration Register (NCR) when an NMI interrupt pin is first enabled, it is possible to get a false interrupt flag. You must follow the following sequence to enable the NMI interrupt on a rising edge event on the NMI pin:

1. Write 1 to NCR[NFEE0] , NCR[NFEE1]
2. Write 1 to NCR[NREE0] , NCR[NREE1]
3. Write 0 to NCR[NFEE0] , NCR[NFEE1]

If the IBE of NMI is tied off to 1, no false interrupt is expected.

45.7 WKPU memory map and registers

45.7.1 WKPU register descriptions

This section provides a detailed description of all the registers accessible in the WKPU module.

NOTE

Reserved registers read as zero and writes have no effect. A transfer error is generated when trying to access a completely reserved register space. The field length of external pad control registers depends on the number of WKPU channels implemented in a chip.

45.7.1.1 WKPU memory map

WKPU base address: 402B_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	NMI Status Flag Register (NSR)	32	RW	0000_0000h
8h	NMI Configuration Register (NCR)	32	RW	6060_0000h
14h	Wakeup/Interrupt Status Flag Register (WISR)	32	RW	0000_0000h
18h	Interrupt Request Enable Register (IRER)	32	RW	0000_0000h
1Ch	Wakeup Request Enable Register (WRER)	32	RW	0000_0000h
28h	Wakeup/Interrupt Rising-Edge Event Enable Register (WIREER)	32	RW	0000_0000h
2Ch	Wakeup/Interrupt Falling-Edge Event Enable Register (WIFEER)	32	RW	0000_0000h
30h	Wakeup/Interrupt Filter Enable Register (WIFER)	32	RW	0000_0000h
54h	Wakeup/Interrupt Status Flag Register (WISR_64)	32	RW	0000_0000h
58h	Interrupt Request Enable Register (IRER_64)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5Ch	Wakeup Request Enable Register (WRER_64)	32	RW	0000_0000h
68h	Wakeup/Interrupt Rising-Edge Event Enable Register (WIREER_64)	32	RW	0000_0000h
6Ch	Wakeup/Interrupt Falling-Edge Event Enable Register (WIFEER_64)	32	RW	0000_0000h
70h	Wakeup/Interrupt Filter Enable Register (WIFER_64)	32	RW	0000_0000h

45.7.1.2 NMI Status Flag Register (NSR)

Offset

Register	Offset
NSR	0h

Function

This register holds the non-maskable interrupt status flags.

NOTE

This register is accessible by 8-, 16-, and 32-bit read/write operations.

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NIF0	NOVF 0	Reserved						NIF1	NOVF 1	Reserved					
W	W1C	W1C							W1C	W1C						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31 NIF0	NMI Status Flag 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This flag can be cleared only by writing 1. Writing 0 has no effect. If enabled (NREE0 or NFEE0 set), NIF0 causes an interrupt request. 0b - No event has occurred on the pad 1b - An event as defined by NREE0 and NFEE0 has occurred
30 NOVF0	NMI Overrun Status Flag 0 This flag can be cleared only by writing 1. Writing 0 has no effect. It will be a copy of the current NIF0 value whenever an NMI event occurs, thereby indicating to the software that the NMI occurred when the last one was not yet serviced. If enabled (NREE0 or NFEE0 set), NOVF0 causes an interrupt request. 0b - No overrun has occurred on NMI input 0 1b - An overrun has occurred on NMI input 0
29-24 —	Reserved
23 NIF1	NMI Status Flag 1 This flag can be cleared only by writing 1. Writing 0 has no effect. If enabled (NREE1 or NFEE1 set), NIF1 causes an interrupt request. 0b - No event has occurred on the pad 1b - An event as defined by NREE1 and NFEE1 has occurred
22 NOVF1	NMI Overrun Status Flag 1 This flag can be cleared only by writing 1. Writing 0 has no effect. It will be a copy of the current NIF1 value whenever an NMI event occurs, thereby indicating to the software that the NMI occurred when the last one was not yet serviced. If enabled (NREE1 or NFEE1 set), NOVF1 causes an interrupt request. 0b - No overrun has occurred on NMI input 1 1b - An overrun has occurred on NMI input 1
21-16 —	Reserved
15-14 —	Reserved
13-8 —	Reserved
7-6 —	Reserved
5-0 —	Reserved

45.7.1.3 NMI Configuration Register (NCR)

Offset

Register	Offset
NCR	8h

Function

This register holds the configuration bits for the non-maskable interrupt settings.

NOTE

- This register is accessible by 8-, 16-, and 32-bit read/write operations.
- Writing 0 to both NREE[*n*] and NFEE[*n*] disables the NMI functionality completely (that is, no non-maskable interrupt is generated on any pad activity).

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NLOCK0	NDSS0		NWRE0	0	NREE0	NFEE0	NFE0	NLOCK1	NDSS1		NWRE1	0	NREE1	NFEE1	NFE1
W																
Reset	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0

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

Fields

Field	Function
31 NLOCK0	NMI Configuration Lock Register 0 Writing 1 to this bit locks the configuration for the NMI until it is unlocked by a system reset or STANDBY0 mode exit. Writing 0 has no effect.
30-29 NDSS0	NMI Destination Source Select 0 **NOTE** As wakeup does not support another interrupt than NMI, the destination source select signal bits are reserved and always retain their reset value. This means no other request other than NMI can be generated. 00b - Non-maskable interrupt 01b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Reserved 11b - Reserved
28 NWRE0	NMI Wakeup Request Enable 0 0b - System wakeup requests from the corresponding NIF0 field are disabled 1b - Causes a system wakeup request when NIF0 = 1 or NOVF0 = 1
27 —	Reserved
26 NREE0	NMI Rising-Edge Events Enable 0 0b - Rising-edge event is disabled 1b - Rising-edge event is enabled
25 NFEE0	NMI Falling-edge Events Enable 0 0b - Falling-edge event is disabled 1b - Falling-edge event is enabled
24 NFE0	NMI Filter Enable 0 Enable analog glitch filter on the NMI pad input. 0b - Filter is disabled 1b - Filter is enabled
23 NLOCK1	NMI Configuration Lock Register 1 Writing 1 to this bit locks the configuration for the NMI until it is unlocked by a system reset or STANDBY0 mode exit . Writing 0 has no effect.
22-21 NDSS1	NMI Destination Source Select 1 **NOTE** As wakeup does not support another interrupt than NMI, the destination source select signal bits are reserved and always retain their reset value. This means no other request other than NMI can be generated. 00b - Non-maskable interrupt 01b - Reserved 10b - Reserved 11b - Reserved
20 NWRE1	NMI Wakeup Request Enable 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - System wakeup requests from the corresponding NIF1 field are disabled 1b - Causes a system wakeup request when NIF1 = 1 or NOVF1 = 1
19 —	Reserved
18 NREE1	NMI Rising-Edge Events Enable 1 0b - Rising-edge event is disabled 1b - Rising-edge event is enabled
17 NFEE1	NMI Falling-Edge Events Enable 1 0b - Falling-edge event is disabled 1b - Falling-edge event is enabled
16 NFE1	NMI Filter Enable 1 Enable analog glitch filter on the NMI pad input. 0b - Filter is disabled 1b - Filter is enabled
15-8 —	Reserved
7-0 —	Reserved

45.7.1.4 Wakeup/Interrupt Status Flag Register (WISR)

Offset

Register	Offset
WISR	14h

Function

This register holds the wakeup/interrupt flags.

NOTE

- This register is accessible only by 32-bit read/write operations.
- Status bits associated with on-chip wakeup sources are located at the left of the external wakeup/interrupt status bits and are read-only. The wakeup for these sources must be configured and cleared at the on-chip wakeup source. Also, the configuration registers for the external interrupts/wakeups do not have corresponding bits.

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EIF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EIF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EIF	External Wakeup/Interrupt Status Flag x This flag can be cleared only by writing 1. Writing 0 has no effect. If enabled (IRER[x]), EIF[x] causes an interrupt request. 0b - No event has occurred on the pad 1b - An event as defined by WIREER and WIFEER has occurred

45.7.1.5 Interrupt Request Enable Register (IRER)

Offset

Register	Offset
IRER	18h

Function

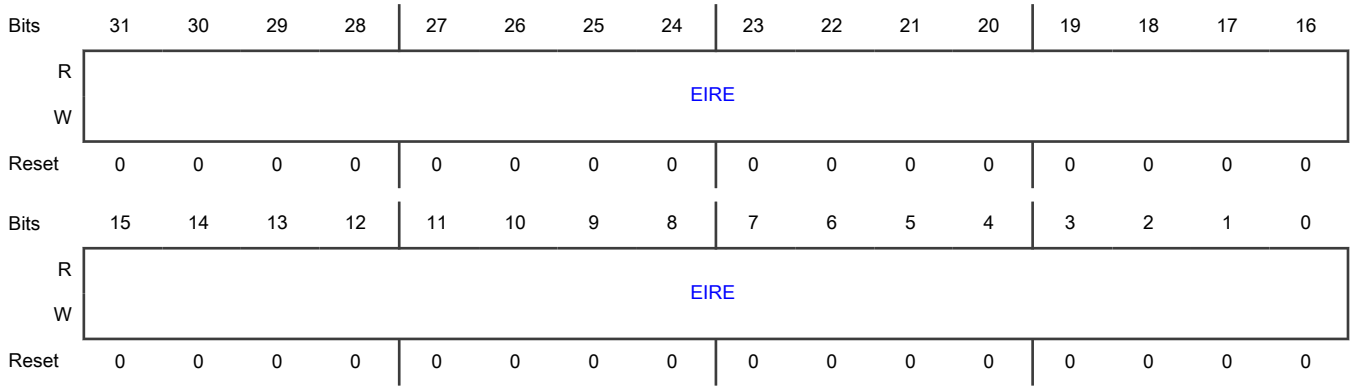
This register is used to enable the interrupt messaging from the wakeup/interrupt pads to the interrupt controller.

NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Diagram



Fields

Field	Function
31-0 EIRE	External Interrupt Request Enable x 0b - Interrupt requests from the corresponding EIF[x] bit are disabled 1b - A set EIF[x] bit causes an interrupt request

45.7.1.6 Wakeup Request Enable Register (WRER)

Offset

Register	Offset
WRER	1Ch

Function

This register is used to enable the system wakeup messaging from the wakeup/interrupt pads to the mode entry and power control modules.

NOTE

This register is accessible only by 32-bit read/write operations.

If a pin is disabled through this register, the corresponding bits in the WIFEER and WIREER registers must be written to 0 to ensure that the pin does not respond to any change.

Access: User read/write

Diagram

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

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

Fields

Field	Function
31-0 WRE	External Wakeup Request Enable x 0b - System wakeup requests from corresponding EIF[x] bit are disabled 1b - A set EIF[x] bit causes a system wakeup request

45.7.1.7 Wakeup/Interrupt Rising-Edge Event Enable Register (WIREER)

Offset

Register	Offset
WIREER	28h

Function

This register is used to enable rising-edge triggered events on the corresponding wakeup/interrupt pads.

NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Diagram

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

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

Fields

Field	Function
31-0 IREE	External Interrupt Rising-edge Events Enable x 0b - Rising-edge event is disabled 1b - Rising-edge event is enabled

45.7.1.8 Wakeup/Interrupt Falling-Edge Event Enable Register (WIFEER)

Offset

Register	Offset
WIFEER	2Ch

Function

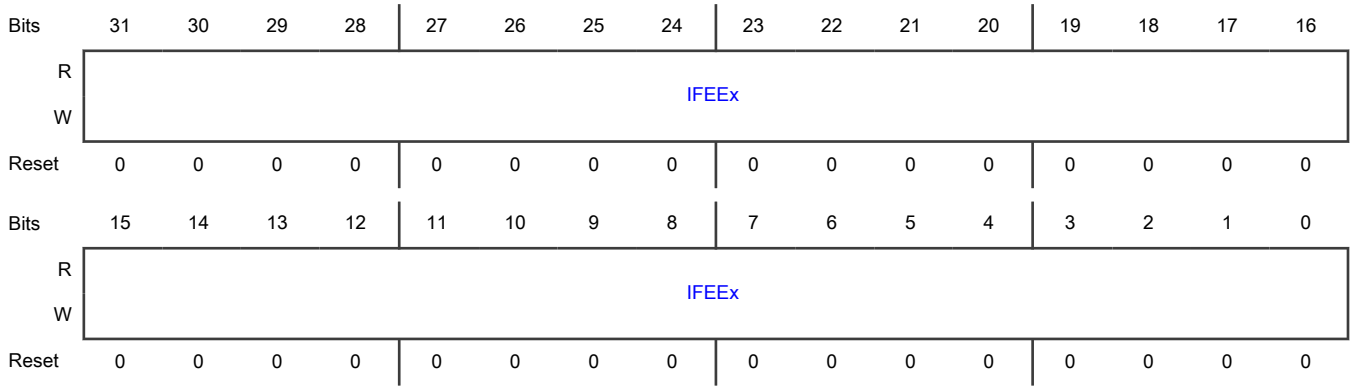
This register is used to enable falling-edge triggered events on the corresponding wakeup/interrupt pads.

NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Diagram



Fields

Field	Function
31-0 IFEEx	External Interrupt Falling-edge Events Enable x 0b - Falling-edge event is disabled 1b - Falling-edge event is enabled

45.7.1.9 Wakeup/Interrupt Filter Enable Register (WIFER)

Offset

Register	Offset
WIFER	30h

Function

This register is used to enable an analog filter on the corresponding interrupt pads to filter out glitches on the inputs.

NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Diagram

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

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

Fields

Field	Function
31-0 IFE	External Interrupt Filter Enable x Enable analog glitch filter on the external interrupt pad input. 0b - Filter is disabled 1b - Filter is enabled

45.7.1.10 Wakeup/Interrupt Status Flag Register (WISR_64)

Offset

Register	Offset
WISR_64	54h

Function

This register holds the wakeup/interrupt flags.

NOTE

- This register is accessible only by 32-bit read/write operations.
- Status bits associated with on-chip wakeup sources are located to the left of the external wakeup/interrupt status bits and are read-only. The wakeup for these sources must be configured and cleared at the on-chip wakeup source. Also, the configuration registers for the external interrupts/wakeups do not have corresponding bits.

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EIF_1															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EIF_1															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EIF_1	External Wakeup/Interrupt Status Flag x This flag can be cleared only by writing 1. Writing 0 has no effect. If enabled (IRER[x]), EIF[x] causes an interrupt request. 0b - No event has occurred on the pad 1b - An event as defined by WIREER and WIFEER has occurred

45.7.1.11 Interrupt Request Enable Register (IRER_64)

Offset

Register	Offset
IRER_64	58h

Function

This register is used to enable interrupt messaging from the wakeup/interrupt pads to the interrupt controller.

NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EIRE_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EIRE_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EIRE_1	External Interrupt Request Enable x 0b - Interrupt requests from the corresponding EIF[x] bit are disabled 1b - A set EIF[x] bit causes an interrupt request

45.7.1.12 Wakeup Request Enable Register (WRER_64)

Offset

Register	Offset
WRER_64	5Ch

Function

This register is used to enable system wakeup messaging from the wakeup/interrupt pads to the mode entry and power control modules.

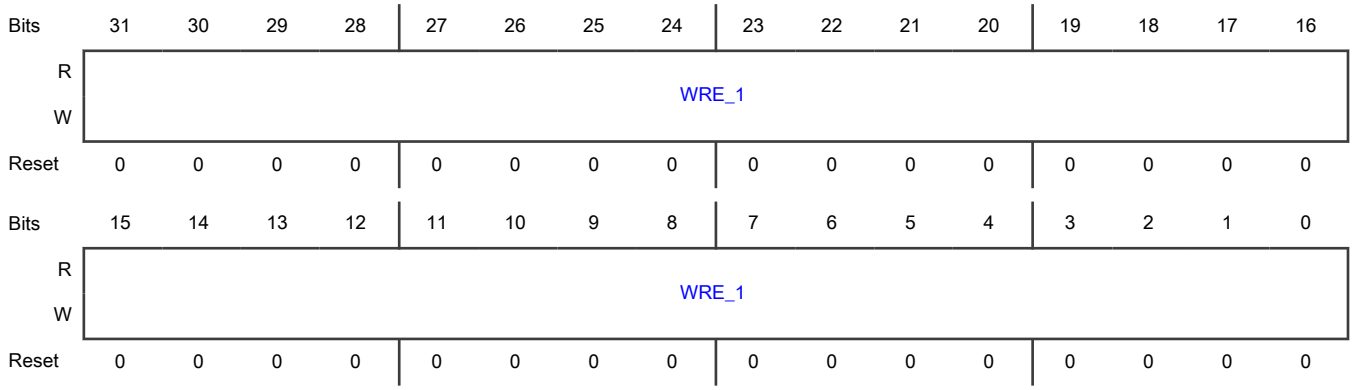
NOTE

This register is accessible only by 32-bit read/write operations.

If a pin is disabled through this register, the corresponding bits in the WIFEER and WIREER registers must be written to 0 to ensure that the pin does not respond to any change.

Access: User read/write

Diagram



Fields

Field	Function
31-0 WRE_1	External Wakeup Request Enable x 0b - System wakeup requests from corresponding EIF[x] bit are disabled 1b - A set EIF[x] bit causes a system wakeup request

45.7.1.13 Wakeup/Interrupt Rising-Edge Event Enable Register (WIREER_64)

Offset

Register	Offset
WIREER_64	68h

Function

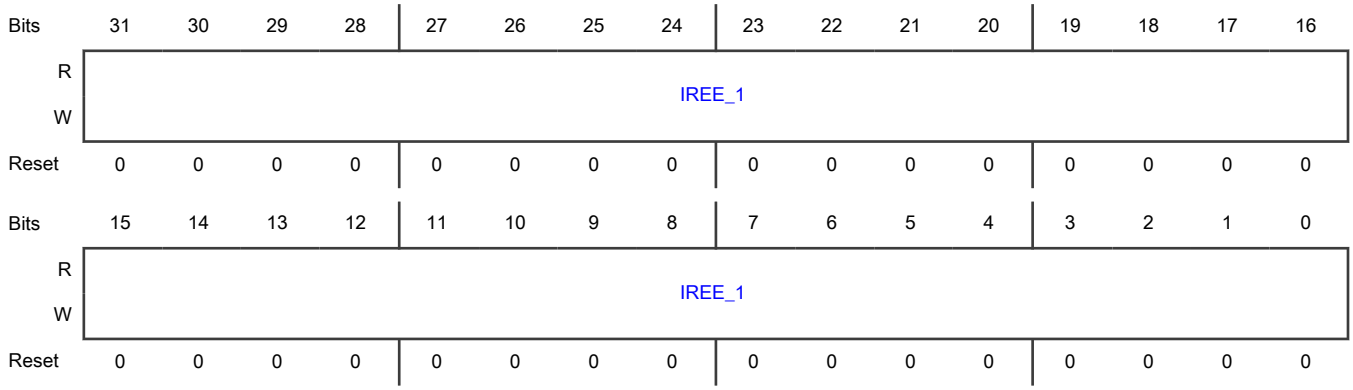
This register is used to enable rising-edge triggered events on the corresponding wakeup/interrupt pads.

NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Diagram



Fields

Field	Function
31-0 IREE_1	External Interrupt Rising-edge Events Enable x 0b - Rising-edge event is disabled 1b - Rising-edge event is enabled

45.7.1.14 Wakeup/Interrupt Falling-Edge Event Enable Register (WIFEER_64)

Offset

Register	Offset
WIFEER_64	6Ch

Function

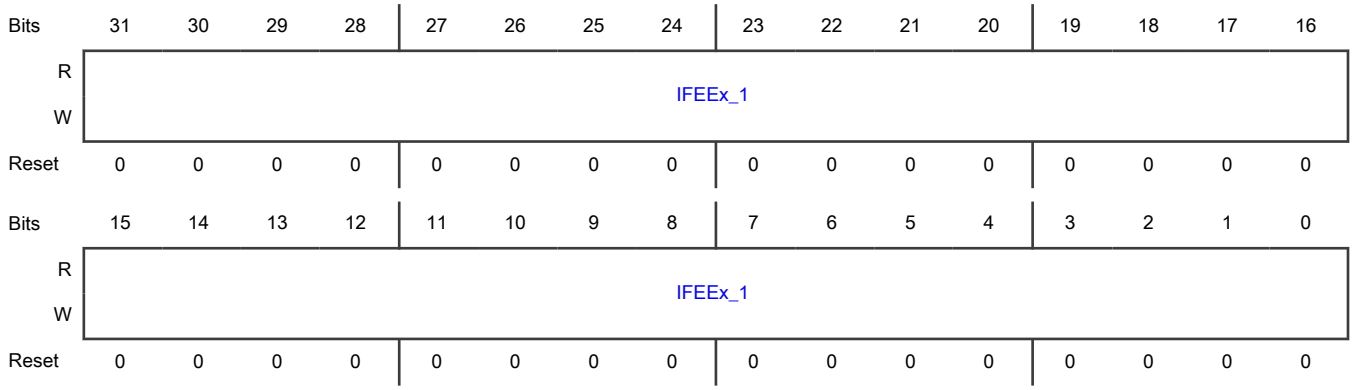
This register is used to enable falling-edge triggered events on the corresponding wakeup/interrupt pads.

NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Diagram



Fields

Field	Function
31-0 IFEEx_1	External Interrupt Falling-edge Events Enable x 0b - Falling-edge event is disabled 1b - Falling-edge event is enabled

45.7.1.15 Wakeup/Interrupt Filter Enable Register (WIFER_64)

Offset

Register	Offset
WIFER_64	70h

Function

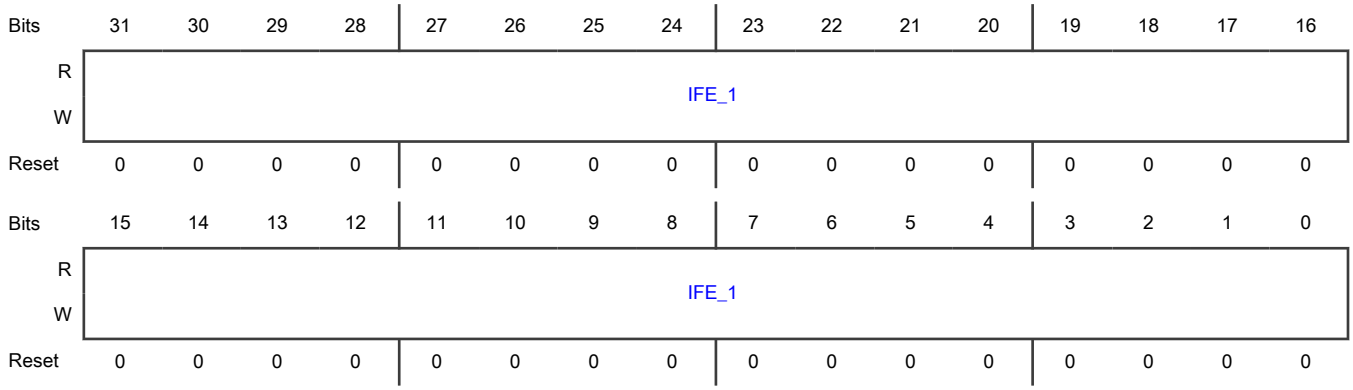
This register is used to enable an analog filter on the corresponding interrupt pads to filter out glitches on the inputs.

NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Diagram



Fields

Field	Function
31-0 IFE_1	External Interrupt Filter Enable x Enable analog glitch filter on the external interrupt pad input. 0b - Filter is disabled 1b - Filter is enabled

45.8 Glossary

NMI Non-maskable interrupts

Chapter 46
Safety Overview

46.1 Introduction

This chip family is developed following the ISO 26262 standards, with derivatives targeting the chips that are operable in a system that fulfills the requirements of the ASIL D or ASIL B safety integrity levels.

Table 220. ASIL levels

Chip	ASIL level
MWCT2D17S, MWCT2D16S, MWCT2016S, MWCT2015S, MWCT2014S	B

The ASIL level targets the safety processing, which means the software functions are executed as intended:

1. Read instructions from memory.
2. Execute instructions.
3. Read data from memory.
4. Process data.
5. Write back the result data into memory.

Some elements of the safety concept are based on the assumption that the chip is connected to an external SBC or PMIC. The SBC or PMIC performs observations and control functionalities that are essential to fulfill some related functional safety requirements. If you do not use an SBC or PMIC, you must ensure that your MWCT-family chip provides an equivalent functionality that properly manages the corresponding interface(s).

The following figure illustrates the safety interface between the chip and the SBC or PMIC.

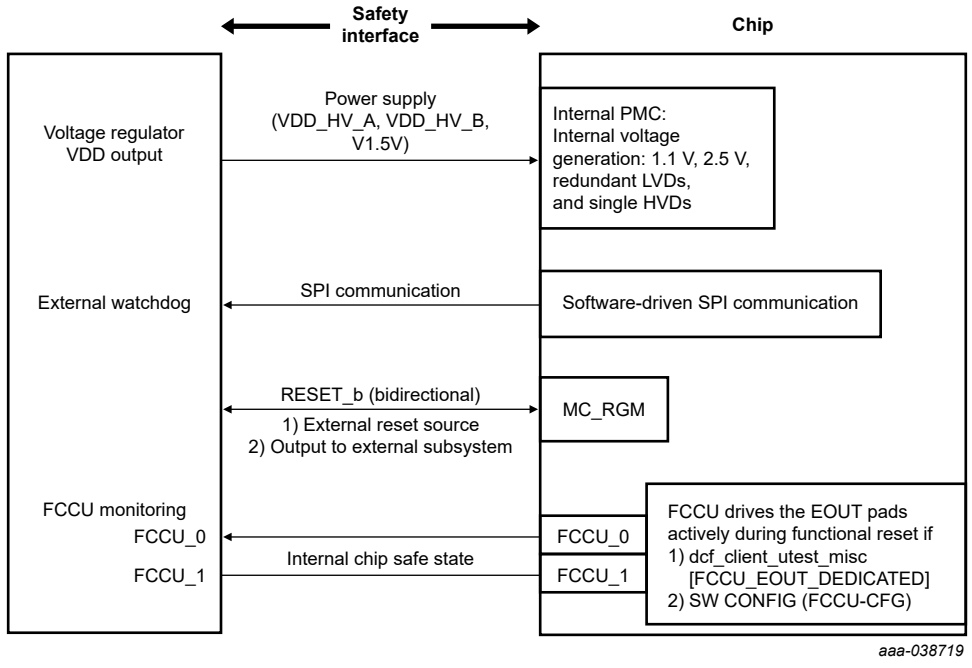


Figure 179. Safety interface between the chip and the SBC or PMIC

In the safety context, the chip interfaces can be classified into the groups shown in the table.

Table 221. Chip interface groups

Interface	Description
Power supply	This interface is between the SBC and the chip. It ensures that the supply to the chip is in the correct range. In case of any low-voltage event, the chip has POR, LVR, and LVD circuits in place and in case of any HVD event, the chip raises an interrupt. The SBC must ensure that the voltage regulator outputs never exceed the allowed range (more specifically for HVD range).
Communication	Responsible for communication between the SBC and the chip. Though the chip supports multiple communication protocols (UART, LIN, SPI, I2C, FlexCAN, and so on), SPI is the preferred communication with SBC. This is relevant to initialize an external watchdog when the chip is inoperative for a considerably long time (this is indicated by the pin states of the communication interface).
Reset	The chip consists of a reset bidirectional pin interfaced with SBC. The SBC can initiate a chip reset via this pin as a safety reaction in case of: • Extreme critical faults • An inoperative chip • Stuck cases based on the criticality and the application requirements The SBC also samples the reset pin state to identify the chip condition (whether in running state or in reset).
FCCU	The chip indicates the chip faults to the SBC via FCCU EOUT pins through this interface.

The interfaces in Table 221 ensure the chip's operational safety and integrity.

46.2 Safety architecture elements

The chip safety architecture consists of following elements that operate in an interconnected way to meet the ASIL requirements:

- Cortex-M7 core complex (including the cache controllers) operating in delayed lockstep (ASIL D only)
- eDMA controller
- Structural core self-test
- Internal windowed watchdogs with independent clock sources
- Power supply monitoring with redundant low-voltage detectors, single high-voltage detectors, and internal ADC connection to check the internal voltages during application
- Robust clock monitoring, including PLL loss of lock detection
- Embedded flash memory with ECC (single-bit data correction, double-bit data detection) and address encoding (parallel address path check)
- System RAM with ECC
  - Single-bit data correction
  - Double-bit data detection

  - Address detection
- Cortex-M7 cache memories with ECC (single-bit data correction, double-bit data detection)
- Peripherals memories (EMAC, FlexCAN) with ECC (single-bit data correction, double-bit data detection)
- End-to-End EDC (E2E EDC), with address encoding and monitoring of the control signals done by a dedicated module (XBIC). This ensures the safety of storage and of the data path to the internal storage (RAM, flash memory, core cache) and peripherals across the crossbar switch. See the XBIC chapter for details.
- Hardware CRC module that supports end-to-end data check integrity for any data transfer in the system (SRAM to peripherals via DMA transfer, external interfaces to SRAM/peripherals, and so on)
- Cortex-M7 MPUs
- XRDC for memory and peripheral protection
- AIPS_Lite peripheral protection with trusted master-slave connection
- Register protection mechanism for safety-critical registers
- On-chip temperature sensor for temperature monitoring
- Self-test:
  - LBIST to detect latent faults in functional logic as well as in safety integrity mechanisms

NOTE

LBIST is not supported in MWCT2016S and MWCT2015S.

  - MBIST to ensure integrity of the memories in the chip (SRAM, ITCM, DTCM, peripheral memories, and so on)
  - Check-the-checker software library
  - ADC self-test
- FCCU for error collection and reaction, including reporting error status to system; FCCU supports these programmable reaction types:
  - Interrupt
  - Functional reset
- Error pads indicate the chip's internal state to the external chip interface or SBC.
- EIM to inject errors into the memories and interface gaskets to verify the error-detection features of the memory controllers and the interface gaskets
- ERM to collect diagnostic information from memory controllers in case of an error event

46.3 I/O peripherals

The arrangement of I/O peripherals across peripheral bridges allows redundant use of peripherals while limiting possible causes of CCF. Redundant use includes using equivalent peripherals in a replicated way as well as using functionally different peripherals in, for example, feedback measurement loops. Comparison of redundant operation is the responsibility of the application software, not the safety hardware mechanism.

The peripherals are distributed evenly across the peripheral bridges (AIPS*n*), except for singular modules like EMAC, QuadSPI, and so on.

NOTE

EMAC and QuadSPI are not present in MWCT2016S and MWCT2015S.

46.4 Self-test

The chip supports a self-test operation. Your safety software must initiate self-test by configuring STCU2; the chip does not initiate it. STCU2 then controls the self-test operation. Self-test supports both MBIST and LBIST. After the self-test operation completes, the chip enters a reset sequence. Self-test results are stored in STCU2 and your safety software can read the results after the reset sequence.

Figure 180 depicts the processing steps related to the self-test operation, and its linkage to the chip reset and an application start. You decide whether to run a self-test or to skip it before starting an application. You must also specify the self-test configuration before relinquishing control to STCU2 for performing the self-test operation. This processing finishes by reentering the chip reset sequence, depending on whether the chip encountered an unrecoverable fault during the self-test operation. When an unrecoverable fault is encountered during the self-test operation, the chip enters a reset sequence by performing a destructive reset. When no such fault is encountered, the self-test operation completes by entering the chip sequence with a functional reset sequence. You can prevent reset cycling by limiting the amount of resets permitted; the chip shuts down when this limit has been reached. See the "Functional reset escalation" and "Destructive reset escalation" sections in the MC_RGM chapter.

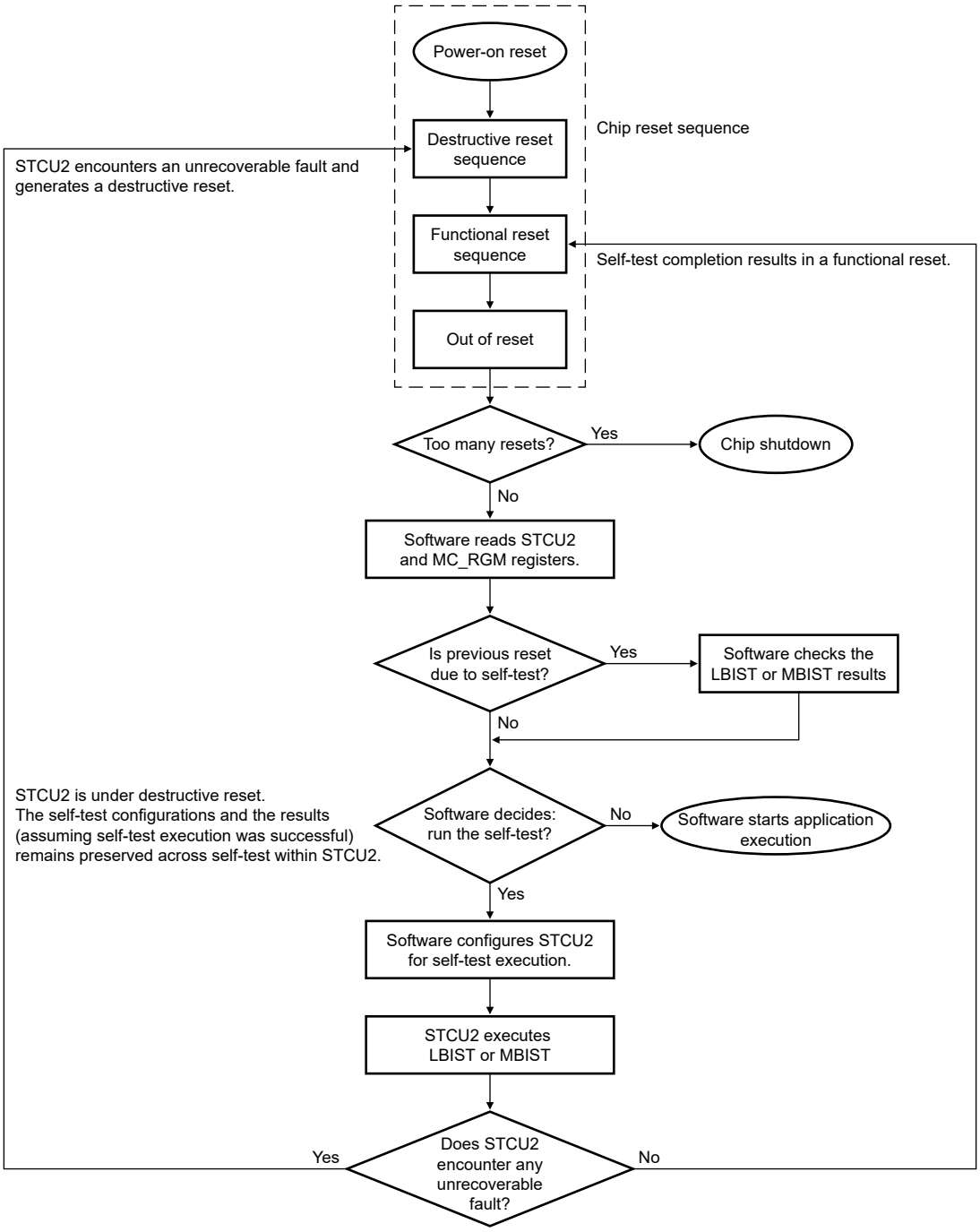


Figure 180. LBIST/MBIST execution

Figure 181 visualizes the top-level partitioning of the chip. The chip consists of two partitions, Run and Standby.

- Run: This partition consists of logic which is present in switchable domain and is shut off (has no supply) while the chip operates in low-power (Standby) mode.
- Standby: This partition consists of always-on logic which is functional even while the chip operates In low-power (Standby) mode.

The Run partition also contains the control logic that is essential for the chip self-test operation, as well as blocks that undergo self-test. The chip consists of a single LBIST partition, which is a subpartition within the Run partition. The LBIST subpartition

contains the logic over which self-test is executed. The logic outside the LBIST subpartition and within the Run partition consists of the LBIST control logic as well as logic which does not undergo LBIST.

The components within these different partitions are listed below the figure (they are not indicated in the figure for clarity).

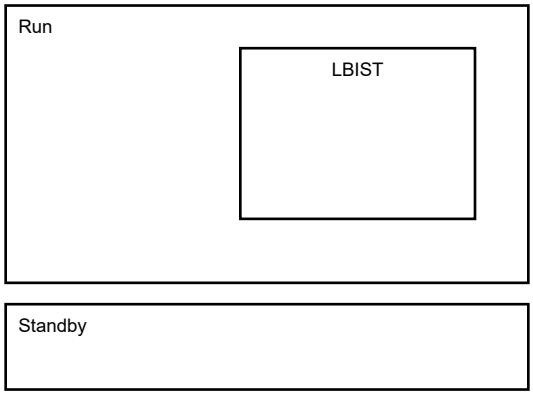


Figure 181. MWCT2D17S chip top-level partition view

The following list specifies the content of the LBIST partition and both power domains related to their participation in the self-test operation:

- Modules participating in the LBIST operation:

  NOTE

  LBIST is not supported in MWCT2016S and MWCT2015S. The modules participating in the LBIST operation are out of LBIST in MWCT2016S and MWCT2015S.

  — eDMA
  — EIM
  — ERM
  — XRDC
    - MDAC0 (Cortex-M7_0)
    - MDAC1 (eDMA)
    - MDAC2 (reserved)
    - MDAC3 (HSE_B)
    - MDAC4 (Cortex-M7_1)
    - MDAC5 (EMAC)
    - MRC0 (flash memory)
    - MRC1 (PRAM)
    - MRC2 (QuadSPI)

      NOTE

      Cortex-M7, EMAC, and QuadSPI are not present in MWCT2016S and MWCT2015S.

    - PDAC1
    - PDAC2
  — CRC
  — FCCU

- FOSU
- PFLASH_CTL
- PRAM_C0
- PRAM_C1
- SWT_1
- CMU
  - CMU_FC_0
  - CMU_FM_1
  - CMU_FM_2
  - CMU_FC_3
  - CMU_FC_4
  - CMU_FC_5
- IAHB gaskets

NOTE

An IAHB gasket is a frequency translation gasket in the system. These gaskets are used to add register walls and synchronize the timing domain across different operating frequencies. See the configurable gaskets in the 'Block diagram' in 'Introduction' chapter for the locations of the IAHB_GASKETs in the chip.

  - DMA_GSKT
  - HSE_GSKT
  - AIPS1_GSKT
  - AIPS2_GSKT
  - QSPI_GSKT

NOTE

AIPS2_GSKT and QSPI_GSKT are not present in MWCT2016S and MWCT2015S.

  - TCM_GSKT
  - EMAC_GSKT
  - EDC master checker gaskets (See block diagram in Introduction chapter for details)
  - EDC slave checker gasket (See block diagram in Introduction chapter for details)
- Crossbar switches and bridges
  - AXBS_0 (main)
  - AXBS_1 (peripheral)
  - AXBS_2 (eDMA)
  - AXBS_3 (Cortex-M7 TCM)
  - AXBS_4 (HSE)
  - AIPS_Lite_1
  - AIPS_Lite_2

NOTE

AIPS2_Lite_ is not present in MWCT2016S and MWCT2015S.

- Modules within the Standby domain (not participating in the LBIST operation):
  - 32 KB standby SRAM
  - LPCMP_0–LPCMP_2
  - Reset generation
    - MC_RGM
    - POR_WDG
  - Power management
    - PMC
    - MC_PCU
    - WKPU
    - Chip standby pins
  - Clock sources
    - FIRC
    - SIRC
    - FXOSC
    - SXOSC

NOTE

SXOSC is not present in MWCT2015S.

  - Timers
    - PIT_0
    - RTC
    - SWT_0
- Modules not participating in the LBIST operation (in the Run domain):
  - HSE subsystem
    - Engine
    - Memories
  - Cortex-M7_0
    - Core
    - I-cache
    - D-cache
    - ITCM
    - DTCM
  - Cortex-M7_1

NOTE

Cortex-M7_1 is not present in MWCT2016S and MWCT2015S.

    - Core
    - I-cache

- D-cache
- ITCM
- DTCM

— CMUs
- CMU0
- CMU3

— Analog blocks
- ADC_0–ADC_2
- TempSense

NOTE

ADC_2 is not present in MWCT2D16S, MWCT2016S, and MWCT2015S.

— Communication modules
- LPSPI_0–LPSPI_5
- LPI2C_0
- LPI2C_1
- LPUART_0–LPUART_15
- FlexCAN_0–FlexCAN_5
- SAI_0
- SAI_1

NOTE

SAI_0 and SAI_1 are not present in MWCT2016S and MWCT2015S.

— Timers
- PIT_1
- PIT_2
- eMIOS
- BCTU
- STM_0
- STM_1

NOTE

PIT_2 and STM_1 are not present in MWCT2016S and MWCT2015S.

— TRGMUX

— PLLDIG

— SIUL2

— LCU_0

— LCU_1

— STCU2

— DMAMUX

— MCM

— MCSM

— INTM

— SEMA42

— MU

NOTE

SEMA42 and chip MU are not present in MWCT2016S and MWCT2015S.

— TSPC

NOTE

REG_PROT of an IP undergoes (or does not undergo) LBIST in conjunction with the protected module that undergoes (or does not undergo) LBIST.

The modules that are vital to the self-test operation are excluded from LBIST regions to allow LBIST to execute successfully.

You must run self-test with PLLDIG configured as the system clock. The LBIST clock controller controls the clock during serial shift, but returns clock control to the functional nodes during the self-test.

46.5 Glossary

ASIL Automotive safety integrity level. This is a risk classification scheme as defined by ISO 26262 for automotive standard.

CCF Common cause failure

DTCM Data tightly coupled memory

ECC Error correction code

ITCM Instruction tightly coupled memory

LBIST Logic built-in self-test

MBIST Memory built-in self-test

PLL Phase-locked loop oscillator

PMIC Power management integrated chip

SBC System basis chip

Chapter 47
Error Injection Module (EIM)

47.1 Chip-specific EIM information

47.1.1 EIM instances

This chip supports up to four instances of EIM:

- EIM_0
- EIM_1
- EIM_2
- EIM_3

Table 222. EIM instances

Instances	MWCT2D16S/MWCT2D17S/MWCT2016S/MWCT2015S/MWCT2014S
EIM_0	Yes
EIM_1	No
EIM_2	No
EIM_3	No

47.1.2 EIM channel mapping

EIM integrates with the memory controller and memory array to enable error injection in a controlled way. Each memory controller has its own EIM channel.

Cortex-M7_1, EMAC, AIPS2 gasket, Cortex-M7 lockstep, and QuadSPI gasket are not available in the MWCT2016S and MWCT2015S product variants of the MWCT family.

Table 223. EIM channel mapping - MWCT2015S, MWCT2x16S, MWCT2D17S

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
0	SRAM0	Word1[31:0] – SRAM0 read data[63:32]	Word0[31:24] – SRAM0 read data ECC[7:0]	64	8
		Word2[31:0] – SRAM0 read data[31:0]			
1	SRAM1[2]	Word1[31:0] – SRAM1 read data[63:32]	Word0[31:24] SRAM1 read data ECC[7:0]	64	8
		Word2[31:0] – SRAM1 read data[31:0]			
2	DMA TCD	Word1[31:0] – DMA TCD RAM read data[63:32]	Word0[31:24] DMA TCD RAM read data checkbits[7:0]	64	8
		Word2[31:0] – DMA TCD RAM read data[31:0]			

Table continues on the next page...

Table 223. EIM channel mapping - MWCT2015S, MWCT2x16S, MWCT2D17S (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
3	Cortex-M7_0 IC tag	Word1[12:0] – Cortex-M7_0 IC tag read data1[28:16]	Word0[31:25] – Cortex-M7_0 IC tag read data1[6:0]	44	14
		Word2[31:22] – Cortex-M7_0 IC tag read data1[15:7]			
		Word2[21:0] – Cortex-M7_0 IC tag read data0[28:7]	Word0[24:18] – Cortex-M7_0 IC tag read data1[6:0]		
4	Cortex-M7_0 IC data	Word1[31:0] – Cortex-M7_0 IC data read data1[71:40]	Word0[31:24] – Cortex-M7_0 IC data data1[7:0]	128	16
		Word2[31:0] – Cortex-M7_0 IC data read data1[39:8]			
		Word3[31:0] – Cortex-M7_0 IC data read data0[71:40]	Word0[23:16] – Cortex-M7_0 IC data read data0[7:0]		
		Word4[31:0] – Cortex-M7_0 IC data read data0[39:8]			
5	Cortex-M7_0 DC tag	Word1[7:0] – Cortex-M7_0 DC tag read data3[32:25]	Word0[31:25] – Cortex-M7_0 DC tag read data3[6:0]	104	28
		Word2[31:14] – Cortex-M7_0 DC tag read data3[24:7]			
		Word2[13:0] – Cortex-M7_0 DC tag read data2[32:19]	Word0[24:18] – Cortex-M7_0 DC tag read data2[6:0]		
		Word3[31:20] – Cortex-M7_0 DC tag read data2[18:7]			
		Word3[19:0] – Cortex-M7_0 DC tag read data1[32:13]	Word0[17:11] – Cortex-M7_0 DC tag read data1[6:0]		
		Word4[31:26] – Cortex-M7_0 DC tag read data1[12:7]			
		Word4[25:0] – Cortex-M7_0 DC tag read data0[32:7]	Word0[10:4] – Cortex-M7_0 DC tag read data0[6:0]		
6	Cortex-M7_0 DC data0	Word1[31:0] – Cortex-M7_0 DC data0 read data3[38:7]	Word0[31:25] – Cortex-M7_0 DC data0 read data3[6:0]	128	28
		Word2[31:0] – Cortex-M7_0 DC data0 read data2[38:7]	Word0[24:18] – Cortex-M7_0 DC data0 read data2[6:0]		
		Word3[31:0] – Cortex-M7_0 DC data0 read data1[38:7]	Word0[17:11] – Cortex-M7_0 DC data0 read data1[6:0]		

Table continues on the next page...

Table 223. EIM channel mapping - MWCT2015S, MWCT2x16S, MWCT2D17S (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
		Word4[31:0] – Cortex-M7_0 DC data0 read data0[38:7]	Word0[10:4] – Cortex-M7_0 DC data0 read data0[6:0]		
7	Cortex-M7_0 DC data1	Word1[31:0] – Cortex-M7_0 DC data1 read data3[38:7]	Word0[31:25] – Cortex-M7_0 DC data1 read data3[6:0]	128	28
		Word2[31:0] – Cortex-M7_0 DC data1 read data2[38:7]	Word0[24:18] – Cortex-M7_0 DC data1 read data2[6:0]		
		Word3[31:0] – Cortex-M7_0 DC data1 read data1[38:7]	Word0[17:11] – Cortex-M7_0 DC data1 read data1[6:0]		
		Word4[31:0] – Cortex-M7_0 DC data1 read data0[38:7]	Word0[10:4] – Cortex-M7_0 DC data1 read data0[6:0]		
8	Cortex-M7_1 IC tag	Word1[12:0] – Cortex-M7_1 IC tag read data1[28:16]	Word0[31:25] – Cortex-M7_1 IC tag read data1[6:0]	44	14
		Word2[31:22] – Cortex-M7_1 IC tag read data1[15:7]			
		Word2[21:0] – Cortex-M7_1 IC tag read data0[28:7]	Word0[24:18] – Cortex-M7_1 IC tag read data0[6:0]		
9	Cortex-M7_1 IC tag	Word1[31:0] – Cortex-M7_1 IC data read data0[71:40]	Word0[31:24] – Cortex-M7_1 IC data read data1[7:0]	128	16
		Word2[31:0] – Cortex-M7_1 IC data read data0[39:8]			
		Word3[31:0] – Cortex-M7_1 IC data read data0[71:40]	Word0[23:16] – Cortex-M7_1 IC data read data0[7:0]		
		Word4[31:0] – Cortex-M7_1 IC data read data0[39:8]			
10	Cortex-M7_1 DC tag	Word1[7:0] – Cortex-M7_1 DC tag read data3[32:25]	Word0[31:25] – Cortex-M7_1 DC tag read data3[6:0]	104	28
		Word2[31:14] – Cortex-M7_1 DC tag read data3[24:7]			
		Word2[31:0] – Cortex-M7_1 DC tag read data2[32:19]	Word0[24:18] – Cortex-M7_1 DC tag read data2[6:0]		
		Word3[31:20] – Cortex-M7_1 DC tag read data1[18:7]			
		Word3[19:0] – Cortex-M7_1 DC tag read data1[32:13]	Word0[17:11] – Cortex-M7_1 DC tag read data1[6:0]		

Table continues on the next page...

Table 223. EIM channel mapping - MWCT2015S, MWCT2x16S, MWCT2D17S (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
		Word4[31:26] – Cortex-M7_1 DC tag read data1[12:7]			
		Word4[25:0] – Cortex-M7_1 DC tag read data0[32:7]	Word0[10:4] – Cortex-M7_1 DC tag read data0[6:0]		
11	Cortex-M7_1 DC data0	Word1[31:0] – Cortex-M7_1 DC data0 read data3[38:7]	Word0[31:25] – Cortex-M7_1 DC data0 read data3[6:0]	128	28
		Word2[31:0] – Cortex-M7_1 DC data0 read data2[38:7]	Word0[24:18] – Cortex-M7_1 DC data0 read data2[6:0]		
		Word3[31:0] – Cortex-M7_1 DC data0 read data1[38:7]	Word0[17:11] – Cortex-M7_1 DC data0 read data1[6:0]		
		Word4[31:0] – Cortex-M7_1 DC data0 read data0[38:7]	Word0[10:4] – Cortex-M7_1 DC data0 read data0[6:0]		
12	Cortex-M7_1 DC data1	Word1[31:0] – Cortex-M7_1 DC data1 read data0[38:7]	Word0[31:25] – Cortex-M7_1 DC data1 read data3[6:0]	128	28
		Word2[31:0] – Cortex-M7_1 DC data1 read data0[38:7]	Word0[24:18] – Cortex-M7_1 DC data1 read data2[6:0]		
		Word3[31:0] – Cortex-M7_1 DC data1 read data0[38:7]	Word0[17:11] – Cortex-M7_1 DC data1 read data1[6:0]		
		Word4[31:0] – Cortex-M7_1 DC data1 read data0[38:7]	Word0[10:4] – Cortex-M7_1 DC data1 read data0[6:0]		
13	Cortex-M7_0 ITCM	Word1[31:0] – Cortex-M7_0 ITCM read data[63:32]	Word0[31:24] – Cortex-M7_0 ITCM read data ECC[7:0]	64	8
		Word2[31:0] – Cortex-M7_0 ITCM read data[31:0]			
14	Cortex-M7_0 D0TCM	Word1[31:0] – Cortex-M7_0 D0TCM read data[31:0]	Word0[31:24] – Cortex-M7_0 D0TCM read data ECC[7:0]	32	8
15	Cortex-M7_0 D1TCM	Word1[31:0] – Cortex-M7_0 D1TCM read data[31:0]	Word0[31:24] – Cortex-M7_0 D1TCM read data ECC[7:0]	32	8
16	Cortex-M7_1 ITCM	Word1[31:0] – Cortex-M7_1 ITCM read data[63:32]	Word0[31:24] – Cortex-M7_1 ITCM read data ECC[7:0]	64	8
		Word2[31:0] – Cortex-M7_1 ITCM read data[31:0]			

Table continues on the next page...

Table 223. EIM channel mapping - MWCT2015S, MWCT2x16S, MWCT2D17S (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
17	Cortex-M7_1 D0TCM	Word1[31:0] – Cortex-M7_1 D0TCM read data[31:0]	Word0[31:24] – Cortex-M7_1 D0TCM read data ECC[7:0]	32	8
18	Cortex-M7_1 D1TCM	Word1[31:0] – Cortex-M7_1 D1TCM read data[31:0]	Word0[31:24] – Cortex-M7_1 D1TCM read data ECC[7:0]	32	8
19	EMAC gasket	Word1[27:0] – EMAC AHB write data[63:36]	—	188	0
		Word2[31:28] – EMAC AHB write data[35:32]			
		Word2[27:0] – EMAC AHB write data[31:4]			
		Word3[31:28] – EMAC AHB write data[3:0]			
		Word3[27:0] – EMAC AHB read data[63:36]			
		Word4[31:28] – EMAC AHB read data[35:32]			
		Word4[27:0] – EMAC AHB read data[31:4]			
		Word5[31:28] – EMAC AHB read data[3:0]			
		Word5[27:0] – EMAC gasket monitor error injection[59:32]			
		Word6[31:0] – EMAC gasket monitor error injection[31:0]			
20	Cortex-M7 TCM gasket	Word1[27:0] – TCM AHB write data[63:36]	—	188	0
		Word2[31:28] – TCM AHB write data[35:32]			
		Word2[27:0] – TCM AHB write data[31:4]			
		Word3[31:28] – TCM AHB write data[3:0]			
		Word3[27:0] – TCM AHB read data[63:36]			
		Word4[31:28] – TCM AHB read data[35:32]			

Table continues on the next page...

Table 223. EIM channel mapping - MWCT2015S, MWCT2x16S, MWCT2D17S (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
		Word4[27:0] – TCM AHB read data[31:4]			
		Word5[31:28] – TCM AHB read data[3:0]			
		Word5[27:0] – TCM gasket monitor error injection[59:32]			
		Word6[31:0] – TCM gasket monitor error injection[31:0]			
21	DMA AXBS S0 gasket	Word1[27:0] – DMA AXBS S0 gasket monitor error injection[59:32]	—	60	0
		Word2[31:0] – DMA AXBS S0 gasket monitor error injection[0:31]			
22	DMA AXBS S1 gasket	Word1[27:0] – DMA AXBS S1 gasket monitor error injection[59:32]	—	60	0
		Word2[31:0] – DMA AXBS S1 gasket monitor error injection[31:0]			
23	HSE gasket	Word1[27:0] – HSE gasket monitor error injection[59:32]	—	60	0
		Word2[31:0] – HSE gasket monitor error injection[31:0]			
24	QuadSPI gasket	Word1[27:0] – QuadSPI gasket monitor error injection[59:32]	—	60	0
		Word2[31:0] – QuadSPI gasket monitor error injection[31:0]			
25	AIPS1 gasket	Word1[27:0] – AIPS1 gasket monitor error injection[59:32]	—	60	0
		Word2[31:0] – AIPS1 gasket monitor error injection[31:0]			
26	AIPS2 gasket	Word1[27:0] – AIPS2 gasket monitor error injection[59:32]	—	60	0

Table continues on the next page...

Table 223. EIM channel mapping - MWCT2015S, MWCT2x16S, MWCT2D17S (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
		Word2[31:0] – AIPS2 gasket monitor error injection[31:0]			
27	Cortex-M7 lockstep	Word1[29:0] – Cortex-M7 error injection[29:0]	—	30	0
28	ECC checking address	Word1[1:0] – Inject error on flash memory controller port 0 address checker	—	24	0
		Word1[3:2] – Inject error on flash memory controller port 1 address checker			
		Word1[4:4] – Inject error on flash memory controller port 2 address checker			
		Word1[7:6] – Inject error on PRAM0 controller address checker			
		Word1[9:8] – Inject error on PRAM1 controller address checker			
		Word1[11:10] – Inject error on 64-bit TCM bus address checker			
		Word1[13:12] – Inject error on QuadSPI path address checker			
		Word1[15:14] – Inject error on AIPS0 address checker			
		Word1[17:16] – Inject error on AIPS1 address checker			
		Word1[19:18] – Inject error on AIPS2 address checker			
		Word1[21:20] – Inject error on 32-bit TCM Cortex-M7_0 path address checker			
		Word1[23:22] – Inject error on 32-bit TCM Cortex-M7_1 path address checker			
		Word1[25:24] – Inject error on DMA AXBS S0 address parity checker[3]			

Table continues on the next page...

Table 223. EIM channel mapping - MWCT2015S, MWCT2x16S, MWCT2D17S (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
		Word1[27:26] – Inject error on DMA AXBS S1 address parity checker[3]			
29	EDC checking wdata	Word1[1:0] – Inject error on PRAM0 controller write data checker	—	18	0
		Word1[3:2] – Inject error on PRAM1 controller write data checker			
		Word1[5:4] – Inject error on 64-bit TCM bus write data checker			
		Word1[7:6] – Reserved			
		Word1[9:8] – Inject error on AIPS0 write data checker			
		Word1[11:10] – Inject error on AIPS1 write data checker			
		Word1[13:12] – Inject error on AIPS2 write data checker			
		Word1[15:14] – Inject error on 32-bit TCM Cortex-M7_0 path write data checker			
		Word1[17:16] – Inject error on 32-bit TCM Cortex-M7_1 path write data checker			
30	EDC checking rdata	Word1[1:0] – Inject error on Cortex-M7_0 AHBM read data checker	—	18	0
		Word1[3:2] – Inject error on Cortex-M7_0 AHBP read data checker			
		Word1[5:4] – Inject error on DMA read data checker			
		Word1[7:6] – Inject error on STAM read data checker			
		Word1[9:8] – Inject error on HSE read data checker			

Table continues on the next page...

Table 223. EIM channel mapping - MWCT2015S, MWCT2x16S, MWCT2D17S (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
		Word1[11:10] – Inject error on EMAC read data checker			
		Word1[13:12] – Inject error on Cortex-M7_1 AHBM read data checker			
		Word1[15:14] – Inject error on Cortex-M7_1 AHBP read data checker			
		Word1[17:16] – Inject error on 32-bit TCM bus path read data checker			

1. You must write to EICHDi_WORDj registers to inject errors in the desired data and check bits. For details, see tables "Error injection channel descriptor: DATA_MASK details" and "DATA_MASK bit: Channel-word mapping" in this chapter.
2. SRAM1 is not available for the MWCT2D16S, MWCT2016S, and MWCT2015S variants.
3. Applicable for MWCT2D16S only.

The two enables, GEIEN and EICHEN*n*, enable the error injection functionality. The former enables it globally and the latter does it for a particular channel. This double-layer enable provides protection against accidental enabling and reconfiguration of the error injection function for each channel.

EIM provides support for inducing single-bit and multi-bit inversions on read data when accessing peripheral RAMs through its data mask registers.

NOTE

For enabling error injection on EDC gaskets (corresponding to channel 28, channel 29, and channel 30), you must also enable the fields corresponding to the required EDC gasket in the MSCM_ENEDC register before enabling the EIM channel.

EIM_EICHD1_CH01, EIM_EICHD1_CH08, EIM_EICHD1_CH09, EIM_EICHD1_CH10, EIM_EICHD1_CH11, EIM_EICHD1_CH12, EIM_EICHD1_CH16, EIM_EICHD1_CH17, EIM_EICHD1_CH18, EIM_EICHD1_CH19, EIM_EICHD1_CH21, EIM_EICHD1_CH22, EIM_EICHD1_CH24, EIM_EICHD1_CH26, EIM_EICHD1_CH27 are not present in MWCT2016S, hence the registers corresponding to these channels are also not present in MWCT2016S.

47.1.3 Behavior of EIM error injection on gaskets

The error injection on gaskets don't impact the actual data flow. The gasket read data, write data and the monitor error don't get changed when EIM error injection is done. The gasket compares the actual data with the modified data (with error injection) and flags the gasket alarm. In case of single-bit error also, the alarm is flagged.

The channels which depict such behavior are the gasket channels listed below:

- Channel 19: GMAC/EMAC gasket
- Channel 20: Cortex-M7 TCM gasket
- Channel 21: DMA AXBS S0 gasket
- Channel 22: DMA AXBS S1 gasket
- Channel 23: HSE gasket
- Channel 24: QuadSPI gasket

- Channel 25: AIPS1 gasket
- Channel 26: AIPS2 gasket

47.2 Overview

The Error Injection Module (EIM) is mainly used for diagnostic purposes. It provides a method for diagnostic coverage of internal memories (for example, system RAM, cache RAMs, and peripheral memories). See the chip-specific EIM information to determine which functional safety features are supported by this method.

EIM enables you to induce artificial errors on error-checking mechanisms of a system, such as ECC for RAM read data and parity bits. For each such mechanism that EIM supports on the chip, EIM can inject single-bit and multi-bit inversions on data in the applicable target bus. Injecting faults on memory accesses can be used to exercise the SEC-DED ECC function of the related system.

47.2.1 Features

The EIM includes these features:

- Supports 31 error injection channels. See the chip-specific EIM information for channel assignment details.
- Protection against accidental enable and reconfiguration error injection function via two-stage enable mechanism

47.2.2 Block diagram

The following diagram shows an example of EIM implementation with a 64-bit read data bus and an 8-bit checkbit bus.

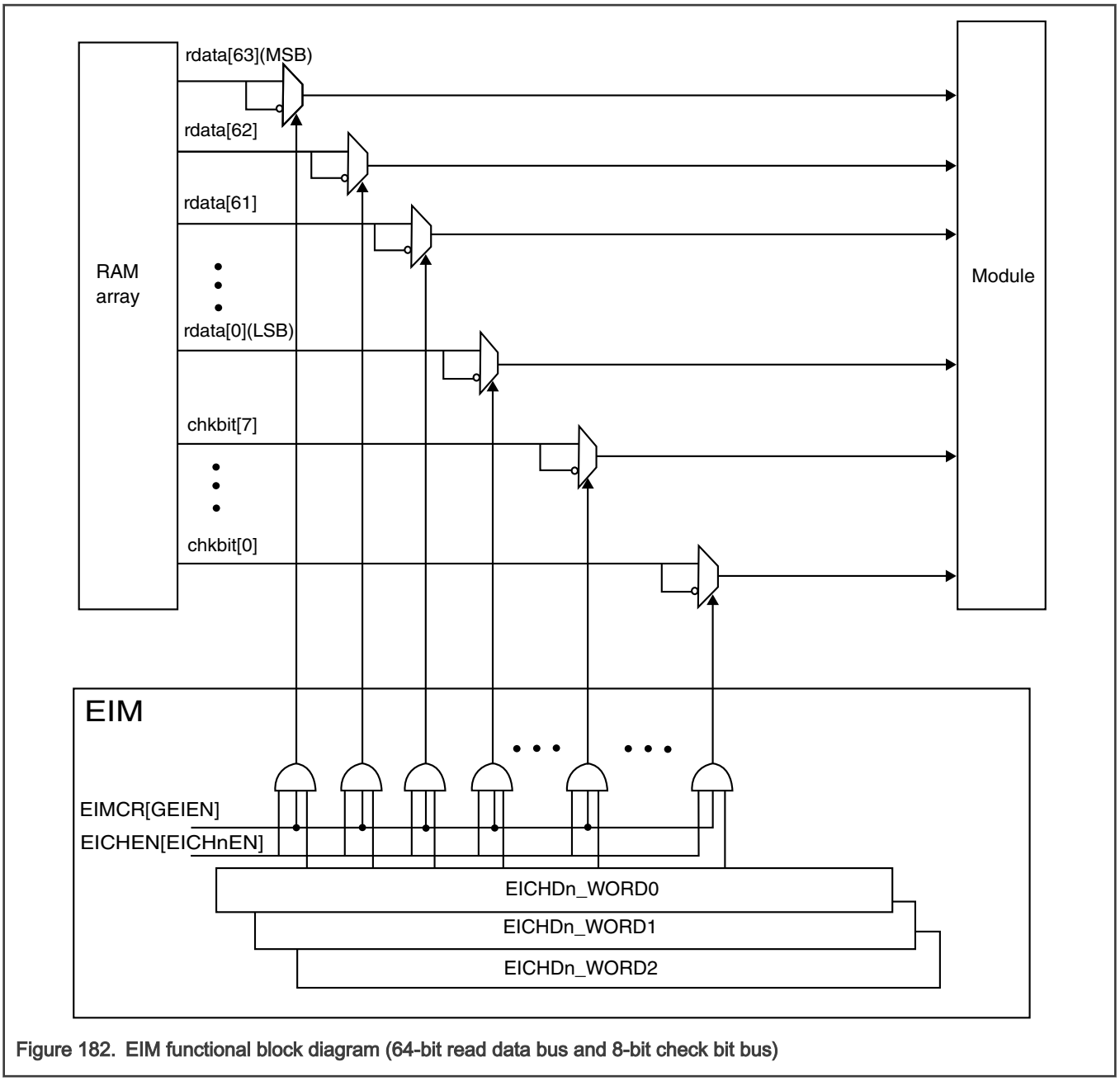


Figure 182. EIM functional block diagram (64-bit read data bus and 8-bit check bit bus)

Several memory elements are implemented within a device, which may not only be the large memory blocks (Flash and SRAM) but also smaller memories like caches, the TCD blocks, and the embedded peripheral memories. Some larger memories may actually be built from multiple memory elements, dependent on their size or function. Each of these memory elements implements its own control logic, the memory controller, that performs the accesses to the actual memory, the memory array. An EIM channel is associated with a memory controller and provides the capability to alter one or multiple signals in the read access path from the corresponding memory array(s). Only memory controllers controlling a safety related memory may be associated with an EIM channel.

47.3 Functional description

The EIM provides protection against accidental enabling and reconfiguration of the error injection function by enforcing a two-stage enablement mechanism. To properly enable the error injection mechanism for a channel:

- Write 1 to the EICHEN[EICH*n*EN] field, where *n* denotes the channel number.

- Write 1 to EIMCR[GEIEN].

NOTE

When the use case for a channel requires writing any EICHD*n*_WORD register, write the EICHD*n*_WORD register before executing the two-stage enablement mechanism. A successful write to any EICHD*n*_WORD register clears the corresponding EICHEN[EICH*n*EN] field.

The EIM supports 31 error injection channels. See the chip-specific EIM information for channel assignment details. Each channel:

- Can be assigned to a single memory array interface by intercepting the assigned memory read data bus and checkbit bus, and injects errors by inverting the value transmitted for selected bits on each bus line.
- Can be assigned to a redundant comparison unit by intercepting the signals being compared, and injecting errors by inverting the value transmitted for selected bits on each bus line.

On a memory read access, the applicable EICHD*n*_WORD registers define which bits of the read data and/or checkbit bus to invert.

Figure 182 depicts the interception and override of a 64-bit read data bus and an 8-bit checkbit data bus for an example memory array.

Error injection scenarios

The EIM supports these cases of error injection:

- To generate a single-bit error, invert only 1 bit of the CHKBIT_MASK or DATA_MASK in the EICHD*n*_WORD registers.
- To generate a multi-bit error, invert only 2 bits of the CHKBIT_MASK or DATA_MASK in the EICHD*n*_WORD registers.

NOTE

An attempt to invert more than 2 bits in one operation might result in undefined behavior.

47.4 EIM register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.

— Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 224. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	64	63-32	31-0	—
1	64	63-32	31-0	—
2	64	63-32	31-0	—
3	44	43-32	31-0	—
4	128	127-96	95-64	63-32
5	104	103-96	95-64	63-32
6	128	127-96	95-64	63-32
7	128	127-96	95-64	63-32
8	44	43-32	31-0	—
9	128	127-96	95-64	63-32
10	104	103-96	95-64	63-32
11	128	127-96	95-64	63-32
12	128	127-96	95-64	63-32
13	64	63-32	31-0	—
14	32	31-0	—	—
15	32	31-0	—	—
16	64	63-32	31-0	—
17	32	31-0	—	—
18	32	31-0	—	—
19	188	187-160	159-128	127-96

Table continues on the next page...

Table 224. Error injection channel descriptor: DATA_MASK details (continued)

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
20	188	187-160	159-128	127-96
21	60	59-32	31-0	—
22	60	59-32	31-0	—
23	60	59-32	31-0	—
24	60	59-32	31-0	—
25	60	59-32	31-0	—
26	60	59-32	31-0	—
27	30	29-0	—	—
28	24	23-0	—	—
29	18	17-0	—	—
30	18	17-0	—	—

Table 225. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
4	31-0	—	—	—	—
5	31-0	—	—	—	—
6	31-0	—	—	—	—
7	31-0	—	—	—	—
9	31-0	—	—	—	—
10	31-0	—	—	—	—
11	31-0	—	—	—	—
12	31-0	—	—	—	—
19	95-64	63-32	31-0	—	—
20	95-64	63-32	31-0	—	—

47.4.1 EIM memory map

EIM base address: 4025_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
140h	Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
180h	Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
1C0h	Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
200h	Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
208h	Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)	32	RW	0000_0000h
20Ch	Error Injection Channel Descriptor 4, Word3 (EICHD4_WORD3)	32	RW	0000_0000h
210h	Error Injection Channel Descriptor 4, Word4 (EICHD4_WORD4)	32	RW	0000_0000h
240h	Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
248h	Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)	32	RW	0000_0000h
24Ch	Error Injection Channel Descriptor 5, Word3 (EICHD5_WORD3)	32	RW	0000_0000h
250h	Error Injection Channel Descriptor 5, Word4 (EICHD5_WORD4)	32	RW	0000_0000h
280h	Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
288h	Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)	32	RW	0000_0000h
28Ch	Error Injection Channel Descriptor 6, Word3 (EICHD6_WORD3)	32	RW	0000_0000h
290h	Error Injection Channel Descriptor 6, Word4 (EICHD6_WORD4)	32	RW	0000_0000h
2C0h	Error Injection Channel Descriptor 7, Word0 (EICHD7_WORD0)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
2C8h	Error Injection Channel Descriptor 7, Word2 (EICHD7_WORD2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2CCh	Error Injection Channel Descriptor 7, Word3 (EICHD7_WORD3)	32	RW	0000_0000h
2D0h	Error Injection Channel Descriptor 7, Word4 (EICHD7_WORD4)	32	RW	0000_0000h
300h	Error Injection Channel Descriptor 8, Word0 (EICHD8_WORD0)	32	RW	0000_0000h
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
308h	Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)	32	RW	0000_0000h
340h	Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
348h	Error Injection Channel Descriptor 9, Word2 (EICHD9_WORD2)	32	RW	0000_0000h
34Ch	Error Injection Channel Descriptor 9, Word3 (EICHD9_WORD3)	32	RW	0000_0000h
350h	Error Injection Channel Descriptor 9, Word4 (EICHD9_WORD4)	32	RW	0000_0000h
380h	Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h
388h	Error Injection Channel Descriptor 10, Word2 (EICHD10_WORD2)	32	RW	0000_0000h
38Ch	Error Injection Channel Descriptor 10, Word3 (EICHD10_WORD3)	32	RW	0000_0000h
390h	Error Injection Channel Descriptor 10, Word4 (EICHD10_WORD4)	32	RW	0000_0000h
3C0h	Error Injection Channel Descriptor 11, Word0 (EICHD11_WORD0)	32	RW	0000_0000h
3C4h	Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)	32	RW	0000_0000h
3C8h	Error Injection Channel Descriptor 11, Word2 (EICHD11_WORD2)	32	RW	0000_0000h
3CCh	Error Injection Channel Descriptor 11, Word3 (EICHD11_WORD3)	32	RW	0000_0000h
3D0h	Error Injection Channel Descriptor 11, Word4 (EICHD11_WORD4)	32	RW	0000_0000h
400h	Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0)	32	RW	0000_0000h
404h	Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)	32	RW	0000_0000h
408h	Error Injection Channel Descriptor 12, Word2 (EICHD12_WORD2)	32	RW	0000_0000h
40Ch	Error Injection Channel Descriptor 12, Word3 (EICHD12_WORD3)	32	RW	0000_0000h
410h	Error Injection Channel Descriptor 12, Word4 (EICHD12_WORD4)	32	RW	0000_0000h
440h	Error Injection Channel Descriptor 13, Word0 (EICHD13_WORD0)	32	RW	0000_0000h
444h	Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)	32	RW	0000_0000h
448h	Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)	32	RW	0000_0000h
480h	Error Injection Channel Descriptor 14, Word0 (EICHD14_WORD0)	32	RW	0000_0000h
484h	Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)	32	RW	0000_0000h
4C0h	Error Injection Channel Descriptor 15, Word0 (EICHD15_WORD0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4C4h	Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)	32	RW	0000_0000h
500h	Error Injection Channel Descriptor 16, Word0 (EICHD16_WORD0)	32	RW	0000_0000h
504h	Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)	32	RW	0000_0000h
508h	Error Injection Channel Descriptor 16, Word2 (EICHD16_WORD2)	32	RW	0000_0000h
540h	Error Injection Channel Descriptor 17, Word0 (EICHD17_WORD0)	32	RW	0000_0000h
544h	Error Injection Channel Descriptor 17, Word1 (EICHD17_WORD1)	32	RW	0000_0000h
580h	Error Injection Channel Descriptor 18, Word0 (EICHD18_WORD0)	32	RW	0000_0000h
584h	Error Injection Channel Descriptor 18, Word1 (EICHD18_WORD1)	32	RW	0000_0000h
5C4h	Error Injection Channel Descriptor 19, Word1 (EICHD19_WORD1)	32	RW	0000_0000h
5C8h	Error Injection Channel Descriptor 19, Word2 (EICHD19_WORD2)	32	RW	0000_0000h
5CCh	Error Injection Channel Descriptor 19, Word3 (EICHD19_WORD3)	32	RW	0000_0000h
5D0h	Error Injection Channel Descriptor 19, Word4 (EICHD19_WORD4)	32	RW	0000_0000h
5D4h	Error Injection Channel Descriptor 19, Word5 (EICHD19_WORD5)	32	RW	0000_0000h
5D8h	Error Injection Channel Descriptor 19, Word6 (EICHD19_WORD6)	32	RW	0000_0000h
604h	Error Injection Channel Descriptor 20, Word1 (EICHD20_WORD1)	32	RW	0000_0000h
608h	Error Injection Channel Descriptor 20, Word2 (EICHD20_WORD2)	32	RW	0000_0000h
60Ch	Error Injection Channel Descriptor 20, Word3 (EICHD20_WORD3)	32	RW	0000_0000h
610h	Error Injection Channel Descriptor 20, Word4 (EICHD20_WORD4)	32	RW	0000_0000h
614h	Error Injection Channel Descriptor 20, Word5 (EICHD20_WORD5)	32	RW	0000_0000h
618h	Error Injection Channel Descriptor 20, Word6 (EICHD20_WORD6)	32	RW	0000_0000h
644h	Error Injection Channel Descriptor 21, Word1 (EICHD21_WORD1)	32	RW	0000_0000h
648h	Error Injection Channel Descriptor 21, Word2 (EICHD21_WORD2)	32	RW	0000_0000h
684h	Error Injection Channel Descriptor 22, Word1 (EICHD22_WORD1)	32	RW	0000_0000h
688h	Error Injection Channel Descriptor 22, Word2 (EICHD22_WORD2)	32	RW	0000_0000h
6C4h	Error Injection Channel Descriptor 23, Word1 (EICHD23_WORD1)	32	RW	0000_0000h
6C8h	Error Injection Channel Descriptor 23, Word2 (EICHD23_WORD2)	32	RW	0000_0000h
704h	Error Injection Channel Descriptor 24, Word1 (EICHD24_WORD1)	32	RW	0000_0000h
708h	Error Injection Channel Descriptor 24, Word2 (EICHD24_WORD2)	32	RW	0000_0000h
744h	Error Injection Channel Descriptor 25, Word1 (EICHD25_WORD1)	32	RW	0000_0000h
748h	Error Injection Channel Descriptor 25, Word2 (EICHD25_WORD2)	32	RW	0000_0000h
784h	Error Injection Channel Descriptor 26, Word1 (EICHD26_WORD1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
788h	Error Injection Channel Descriptor 26, Word2 (EICHD26_WORD2)	32	RW	0000_0000h
7C4h	Error Injection Channel Descriptor 27, Word1 (EICHD27_WORD1)	32	RW	0000_0000h
804h	Error Injection Channel Descriptor 28, Word1 (EICHD28_WORD1)	32	RW	0000_0000h
844h	Error Injection Channel Descriptor 29, Word1 (EICHD29_WORD1)	32	RW	0000_0000h
884h	Error Injection Channel Descriptor 30, Word1 (EICHD30_WORD1)	32	RW	0000_0000h

47.4.2 Error Injection Module Configuration Register (EIMCR)

Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram

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

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

Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled

47.4.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	EICH5EN	EICH6EN	EICH7EN	EICH8EN	EICH9EN	EICH10EN	EICH11EN	EICH12EN	EICH13EN	EICH14EN	EICH15EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EICH16EN	EICH17EN	EICH18EN	EICH19EN	EICH20EN	EICH21EN	EICH22EN	EICH23EN	EICH24EN	EICH25EN	EICH26EN	EICH27EN	EICH28EN	EICH29EN	EICH30EN	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10
20 EICH11EN	Error Injection Channel 11 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Error injection is disabled on Error Injection Channel 11 1b - Error injection is enabled on Error Injection Channel 11
19 EICH12EN	Error Injection Channel 12 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 12 1b - Error injection is enabled on Error Injection Channel 12
18 EICH13EN	Error Injection Channel 13 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 13 1b - Error injection is enabled on Error Injection Channel 13
17 EICH14EN	Error Injection Channel 14 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 14 1b - Error injection is enabled on Error Injection Channel 14
16 EICH15EN	Error Injection Channel 15 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 15 1b - Error injection is enabled on Error Injection Channel 15
15 EICH16EN	Error Injection Channel 16 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 16 1b - Error injection is enabled on Error Injection Channel 16
14 EICH17EN	Error Injection Channel 17 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 17 1b - Error injection is enabled on Error Injection Channel 17
13 EICH18EN	Error Injection Channel 18 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 18 1b - Error injection is enabled on Error Injection Channel 18

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 EICH19EN	Error Injection Channel 19 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 19 1b - Error injection is enabled on Error Injection Channel 19
11 EICH20EN	Error Injection Channel 20 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 20 1b - Error injection is enabled on Error Injection Channel 20
10 EICH21EN	Error Injection Channel 21 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 21 1b - Error injection is enabled on Error Injection Channel 21
9 EICH22EN	Error Injection Channel 22 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Error injection is disabled on Error Injection Channel 22 1b - Error injection is enabled on Error Injection Channel 22
8 EICH23EN	Error Injection Channel 23 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 23 1b - Error injection is enabled on Error Injection Channel 23
7 EICH24EN	Error Injection Channel 24 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 24 1b - Error injection is enabled on Error Injection Channel 24
6 EICH25EN	Error Injection Channel 25 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 25 1b - Error injection is enabled on Error Injection Channel 25
5 EICH26EN	Error Injection Channel 26 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 26 1b - Error injection is enabled on Error Injection Channel 26
4 EICH27EN	Error Injection Channel 27 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 27 1b - Error injection is enabled on Error Injection Channel 27
3 EICH28EN	Error Injection Channel 28 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 28 1b - Error injection is enabled on Error Injection Channel 28
2 EICH29EN	Error Injection Channel 29 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 29 1b - Error injection is enabled on Error Injection Channel 29

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 EICH30EN	Error Injection Channel 30 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 30 1b - Error injection is enabled on Error Injection Channel 30
0 —	Reserved

47.4.4 Error Injection Channel Descriptor n, Word0 (EICHD0_WORD0 - EICHD2_WORD0)

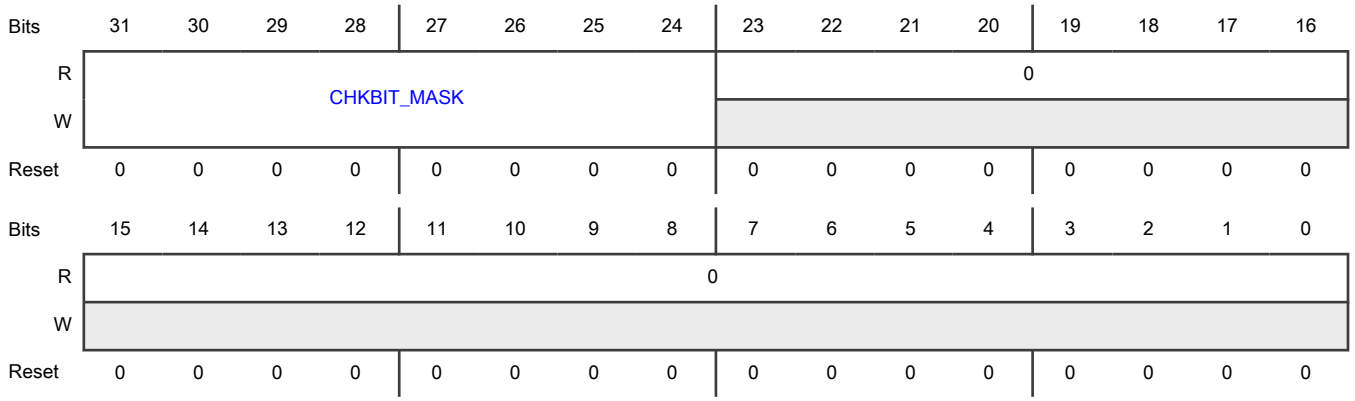
Offset

Register	Offset
EICHD0_WORD0	100h
EICHD1_WORD0	140h
EICHD2_WORD0	180h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

47.4.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD2_WORD1)

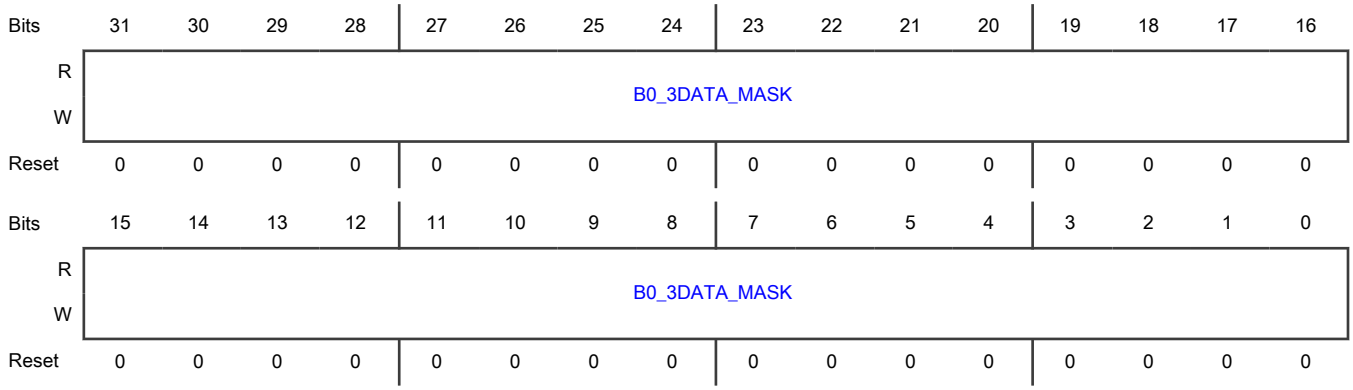
Offset

Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h
EICHD2_WORD1	184h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.4.6 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD26_WORD2)

Offset

Register	Offset
EICHD0_WORD2	108h
EICHD1_WORD2	148h
EICHD2_WORD2	188h
EICHD3_WORD2	1C8h
EICHD4_WORD2	208h
EICHD5_WORD2	248h
EICHD6_WORD2	288h
EICHD7_WORD2	2C8h
EICHD8_WORD2	308h
EICHD9_WORD2	348h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
EICHD10_WORD2	388h
EICHD11_WORD2	3C8h
EICHD12_WORD2	408h
EICHD13_WORD2	448h
EICHD16_WORD2	508h
EICHD19_WORD2	5C8h
EICHD20_WORD2	608h
EICHD21_WORD2	648h
EICHD22_WORD2	688h
EICHD23_WORD2	6C8h
EICHD24_WORD2	708h
EICHD25_WORD2	748h
EICHD26_WORD2	788h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0	Data Mask Bytes 4-7

Table continues on the next page...

Field	Function
B4_7DATA_MASK	This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

47.4.7 Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)

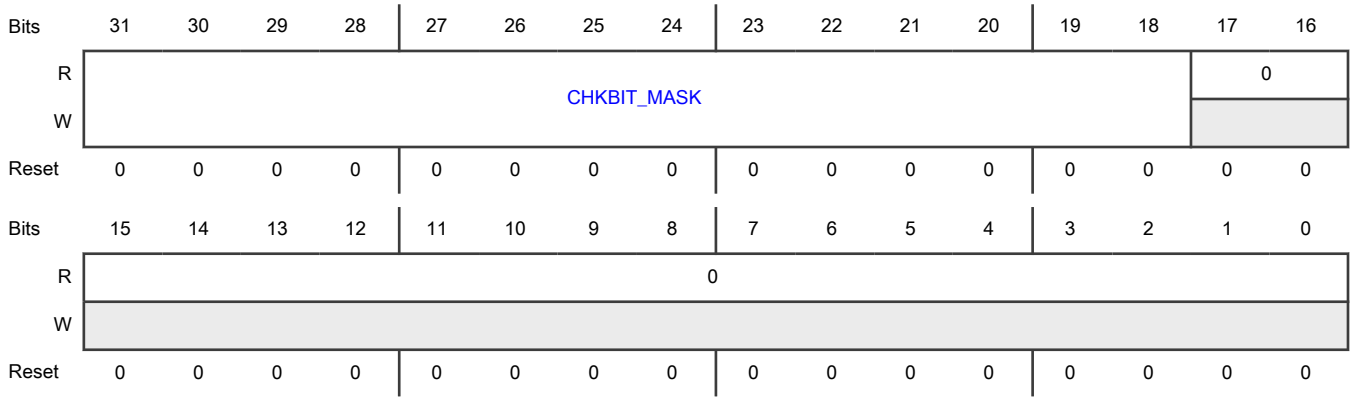
Offset

Register	Offset
EICHD3_WORD0	1C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-18 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[13:0] (14 bits wide), CHKBIT_MASK[13] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
17-0 —	Reserved

47.4.8 Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)

Offset

Register	Offset
EICHD3_WORD1	1C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				B0_3DATA_MASK											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.4.9 Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0)

Offset

Register	Offset
EICHD4_WORD0	200h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram

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

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

Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

47.4.10 Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)

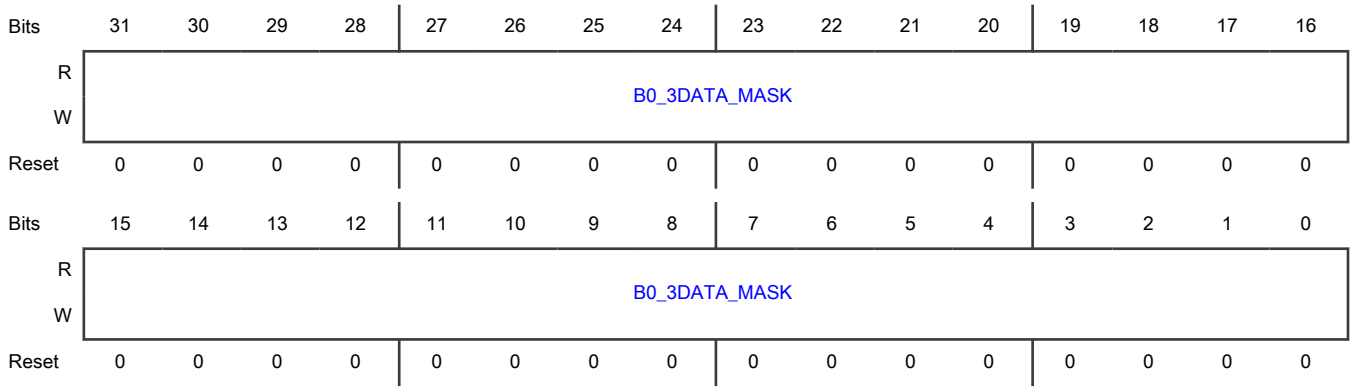
Offset

Register	Offset
EICHD4_WORD1	204h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.4.11 Error Injection Channel Descriptor n, Word3 (EICHD4_WORD3 - EICHD20_WORD3)

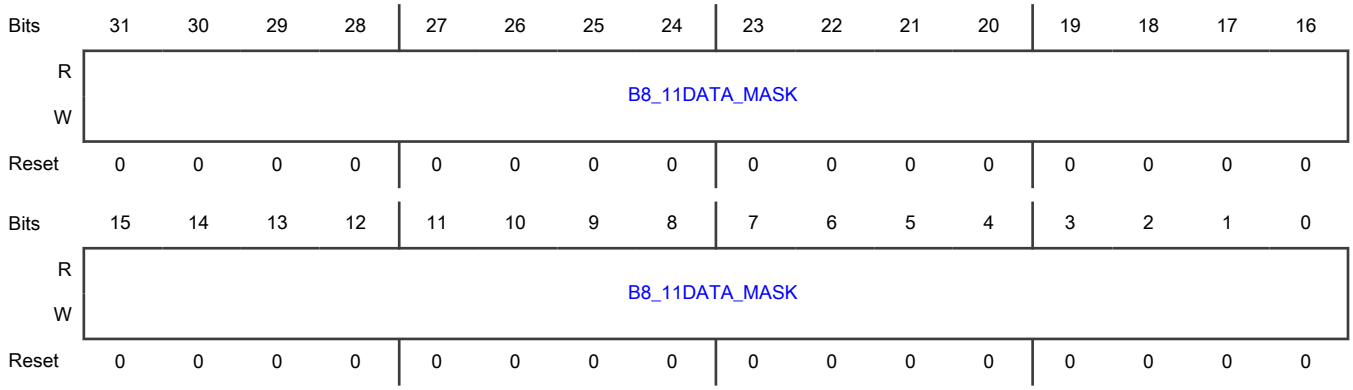
Offset

Register	Offset
EICHD4_WORD3	20Ch
EICHD5_WORD3	24Ch
EICHD6_WORD3	28Ch
EICHD7_WORD3	2CCh
EICHD9_WORD3	34Ch
EICHD10_WORD3	38Ch
EICHD11_WORD3	3CCh
EICHD12_WORD3	40Ch
EICHD19_WORD3	5CCh
EICHD20_WORD3	60Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Bytes 8-11 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

47.4.12 Error Injection Channel Descriptor n, Word4 (EICHD4_WORD4 - EICHD20_WORD4)

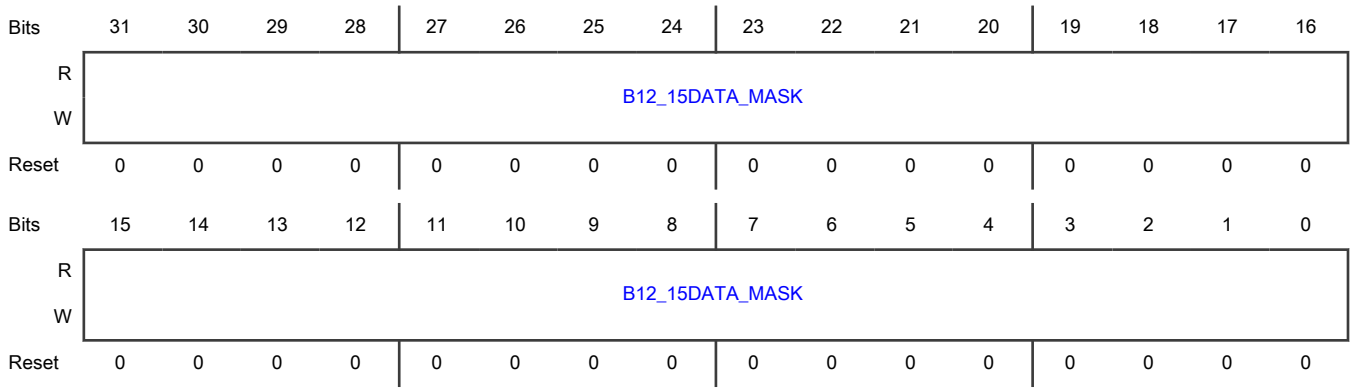
Offset

Register	Offset
EICHD4_WORD4	210h
EICHD5_WORD4	250h
EICHD6_WORD4	290h
EICHD7_WORD4	2D0h
EICHD9_WORD4	350h
EICHD10_WORD4	390h
EICHD11_WORD4	3D0h
EICHD12_WORD4	410h
EICHD19_WORD4	5D0h
EICHD20_WORD4	610h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

47.4.13 Error Injection Channel Descriptor n, Word0 (EICHD5_WORD0 - EICHD7_WORD0)

Offset

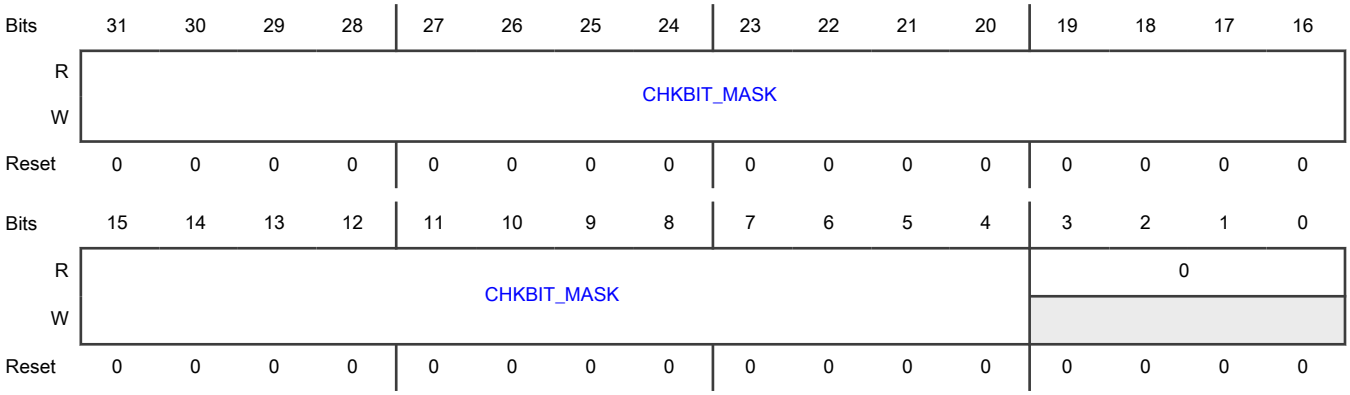
Register	Offset
EICHD5_WORD0	240h
EICHD6_WORD0	280h
EICHD7_WORD0	2C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or

remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-4 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[27:0] (28 bits wide), CHKBIT_MASK[27] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
3-0 —	Reserved

47.4.14 Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)

Offset

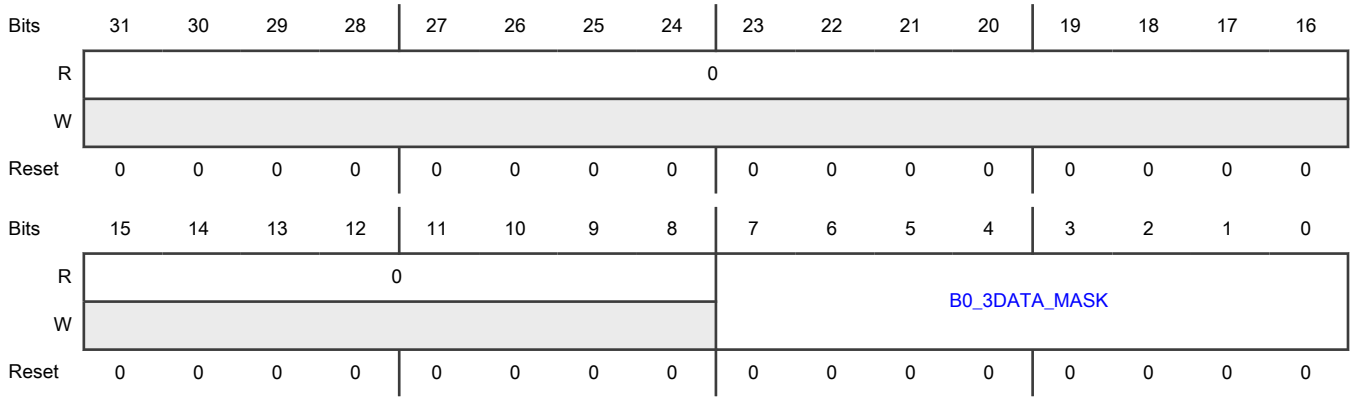
Register	Offset
EICHD5_WORD1	244h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted

or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.4.15 Error Injection Channel Descriptor n, Word1 (EICHD6_WORD1 - EICHD7_WORD1)

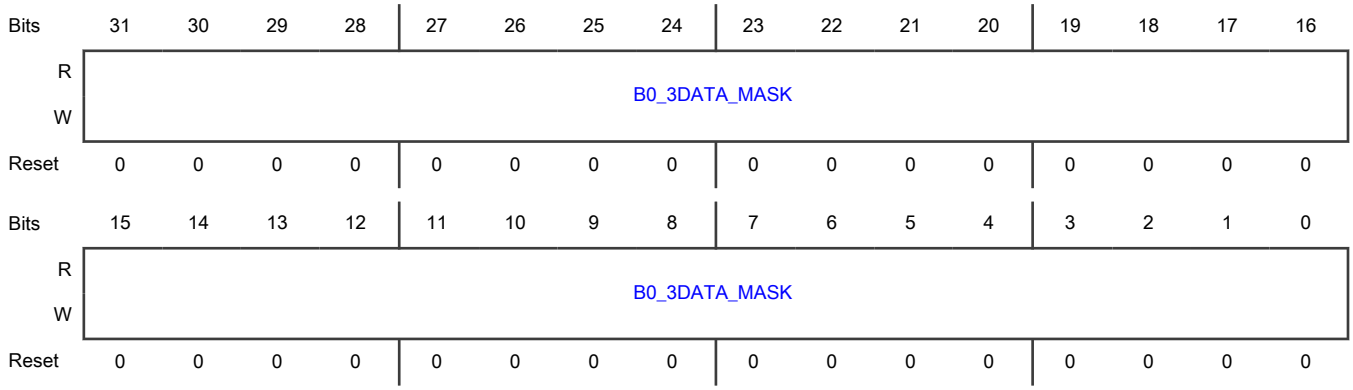
Offset

Register	Offset
EICHD6_WORD1	284h
EICHD7_WORD1	2C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.4.16 Error Injection Channel Descriptor 8, Word0 (EICHD8_WORD0)

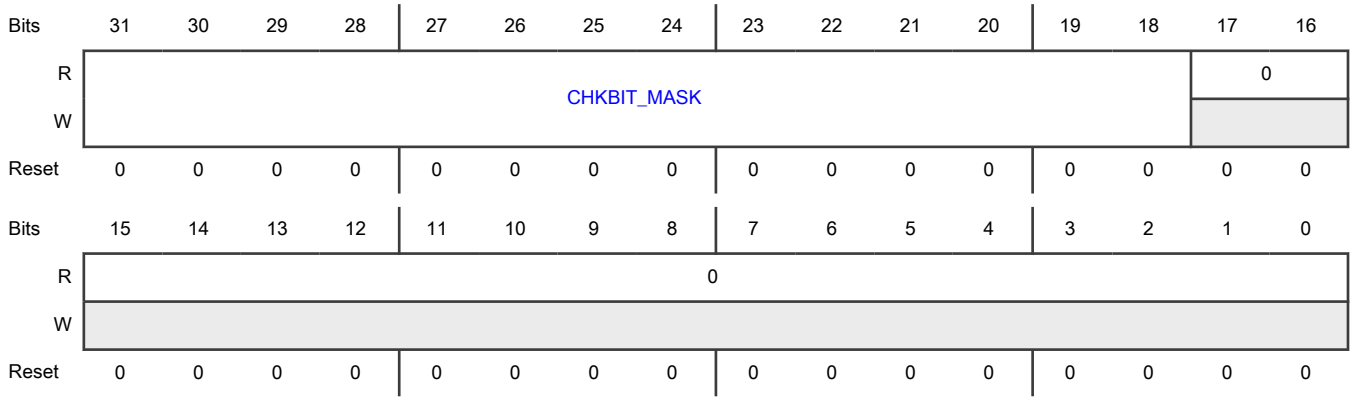
Offset

Register	Offset
EICHD8_WORD0	300h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-18 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[13:0] (14 bits wide), CHKBIT_MASK[13] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
17-0 —	Reserved

47.4.17 Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)

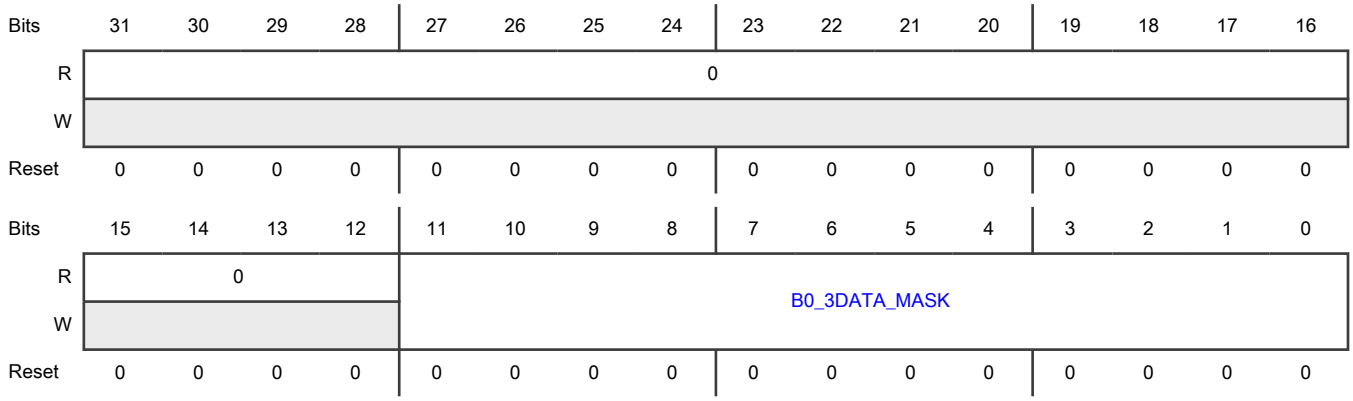
Offset

Register	Offset
EICHD8_WORD1	304h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.4.18 Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0)

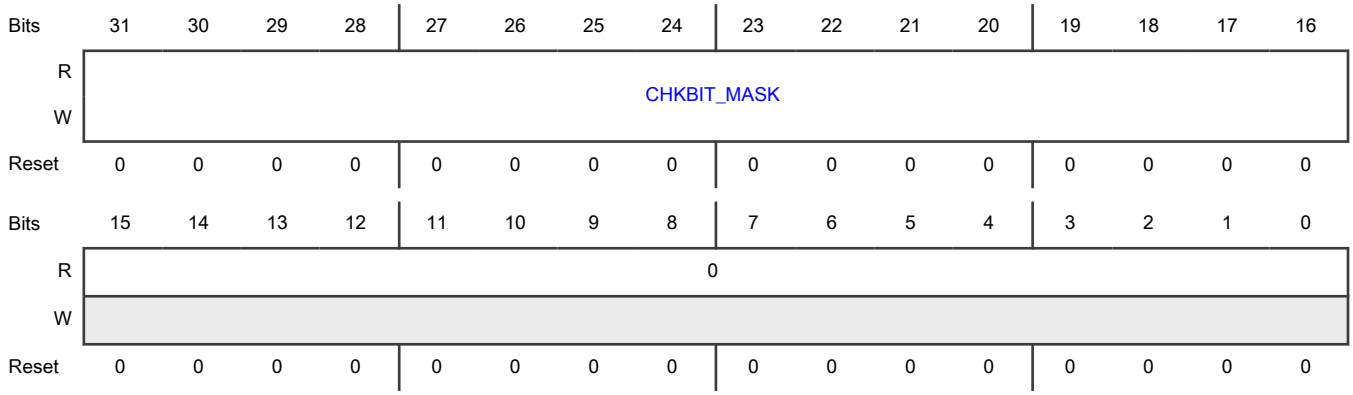
Offset

Register	Offset
EICHD9_WORD0	340h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

47.4.19 Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)

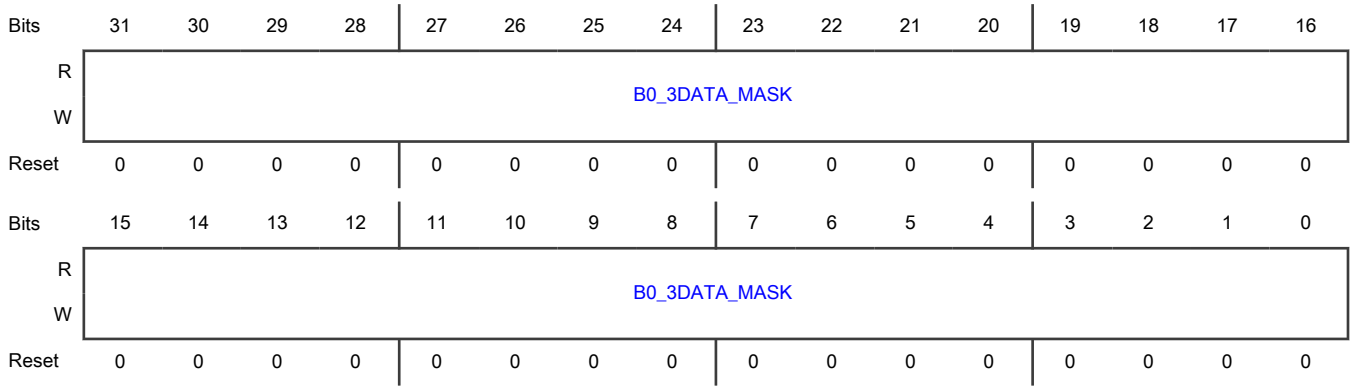
Offset

Register	Offset
EICHD9_WORD1	344h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.4.20 Error Injection Channel Descriptor n, Word0 (EICHD10_WORD0 - EICHD12_WORD0)

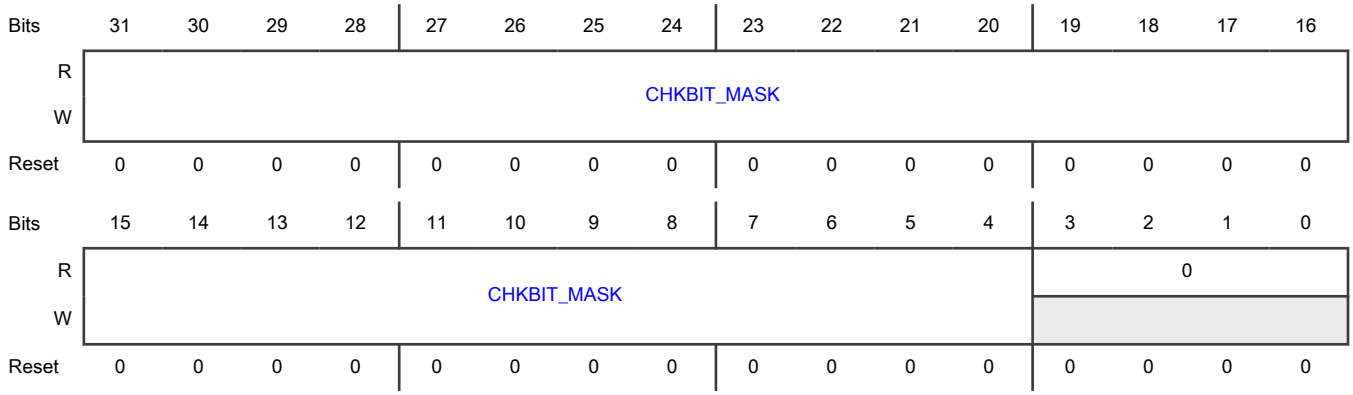
Offset

Register	Offset
EICHD10_WORD0	380h
EICHD11_WORD0	3C0h
EICHD12_WORD0	400h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-4 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[27:0] (28 bits wide), CHKBIT_MASK[27] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
3-0 —	Reserved

47.4.21 Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)

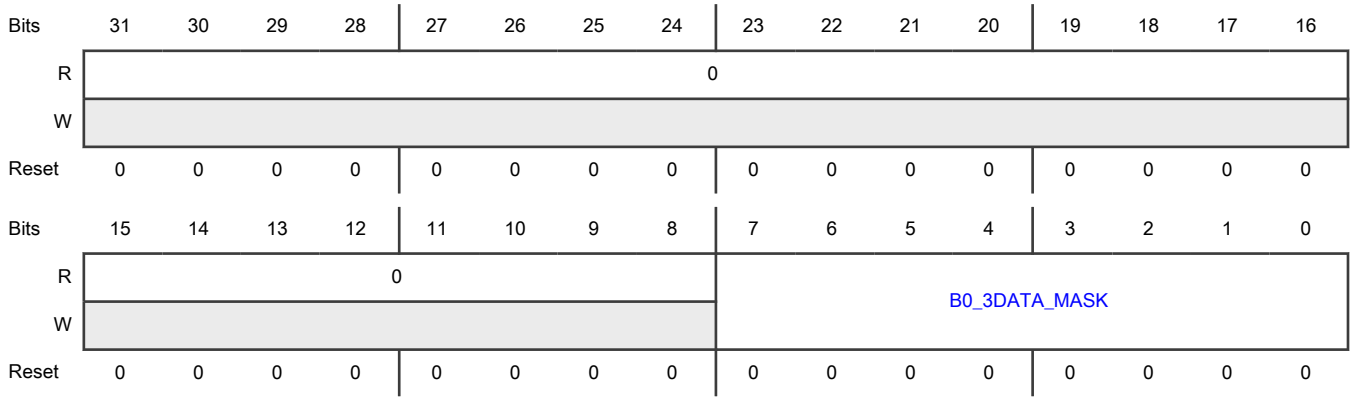
Offset

Register	Offset
EICHD10_WORD1	384h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.4.22 Error Injection Channel Descriptor n, Word1 (EICHD11_WORD1 - EICHD18_WORD1)

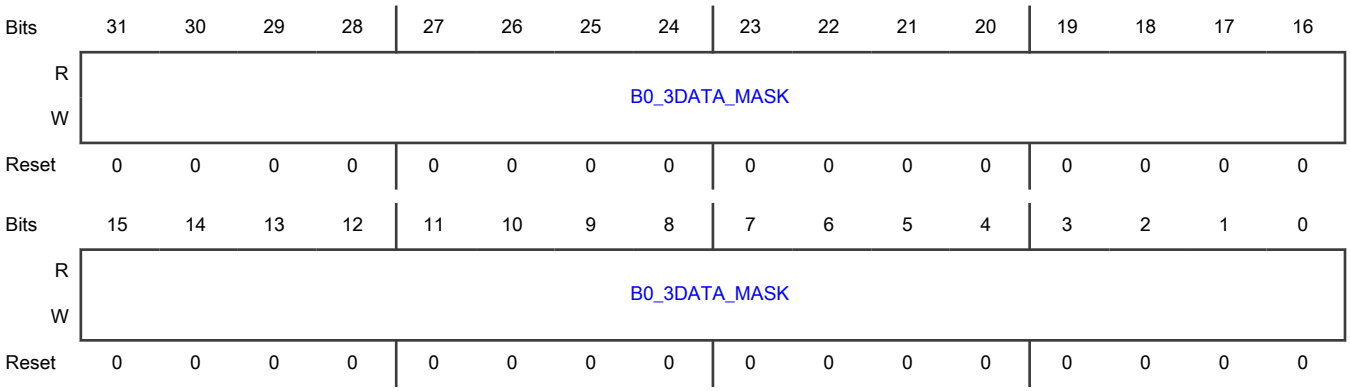
Offset

Register	Offset
EICHD11_WORD1	3C4h
EICHD12_WORD1	404h
EICHD13_WORD1	444h
EICHD14_WORD1	484h
EICHD15_WORD1	4C4h
EICHD16_WORD1	504h
EICHD17_WORD1	544h
EICHD18_WORD1	584h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.4.23 Error Injection Channel Descriptor n, Word0 (EICHD13_WORD0 - EICHD18_WORD0)

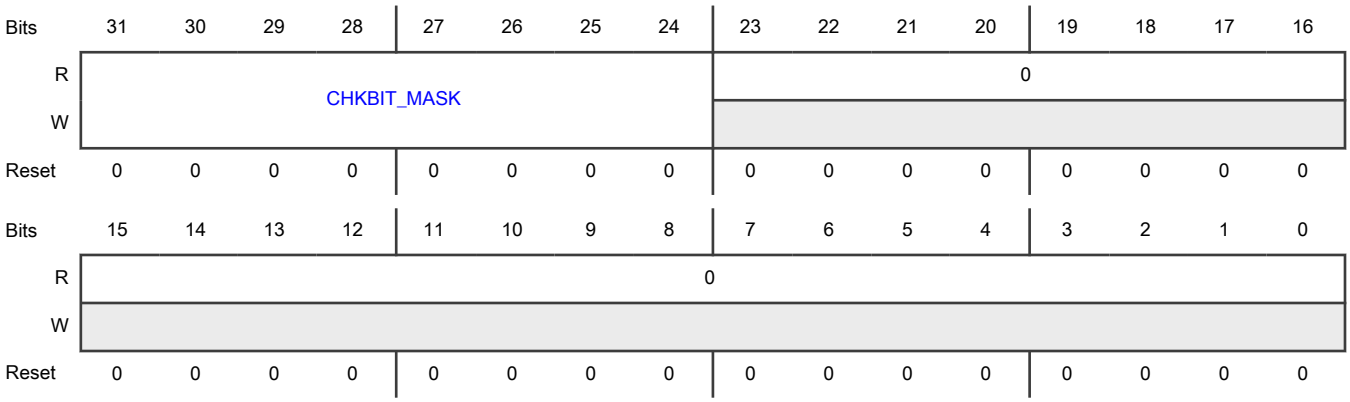
Offset

Register	Offset
EICHD13_WORD0	440h
EICHD14_WORD0	480h
EICHD15_WORD0	4C0h
EICHD16_WORD0	500h
EICHD17_WORD0	540h
EICHD18_WORD0	580h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

47.4.24 Error Injection Channel Descriptor n, Word1 (EICHD19_WORD1 - EICHD26_WORD1)

Offset

Register	Offset
EICHD19_WORD1	5C4h

Table continues on the next page...

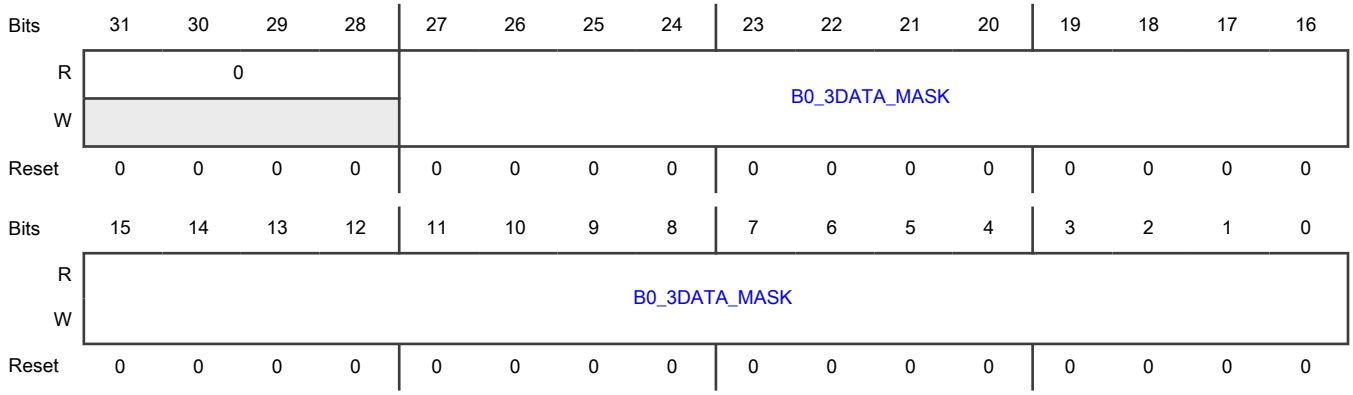
Table continued from the previous page...

Register	Offset
EICHD20_WORD1	604h
EICHD21_WORD1	644h
EICHD22_WORD1	684h
EICHD23_WORD1	6C4h
EICHD24_WORD1	704h
EICHD25_WORD1	744h
EICHD26_WORD1	784h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.4.25 Error Injection Channel Descriptor n, Word5 (EICHD19_WORD5 - EICHD20_WORD5)

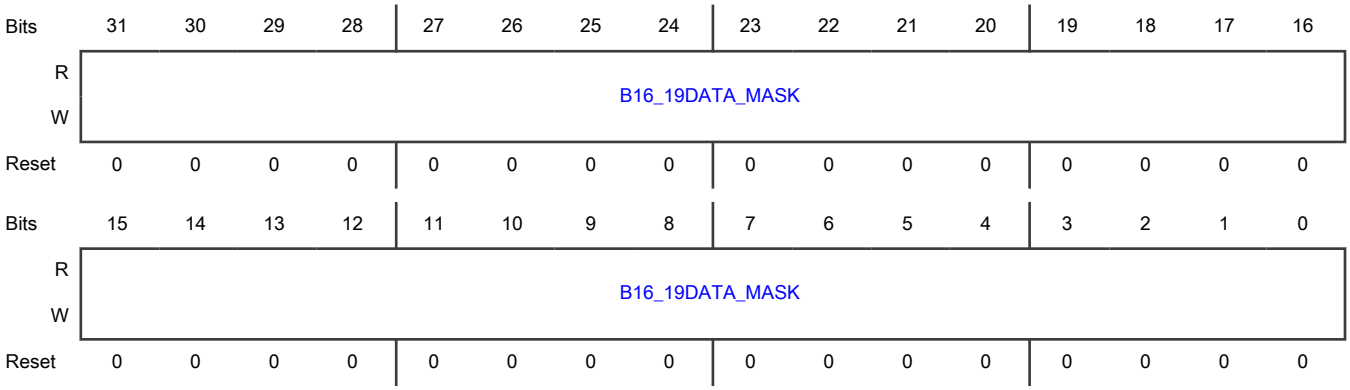
Offset

Register	Offset
EICHD19_WORD5	5D4h
EICHD20_WORD5	614h

Function

The sixth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B16_19DATA_MASK correspond to bytes 16–19 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B16_19DATA_MASK	Data Mask Bytes 16-19 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B16_19DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details.

Table continues on the next page...

Field	Function
	0b - The corresponding bit of bytes 16-19 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 16-19 on the read data bus is inverted.

47.4.26 Error Injection Channel Descriptor n, Word6 (EICHD19_WORD6 - EICHD20_WORD6)

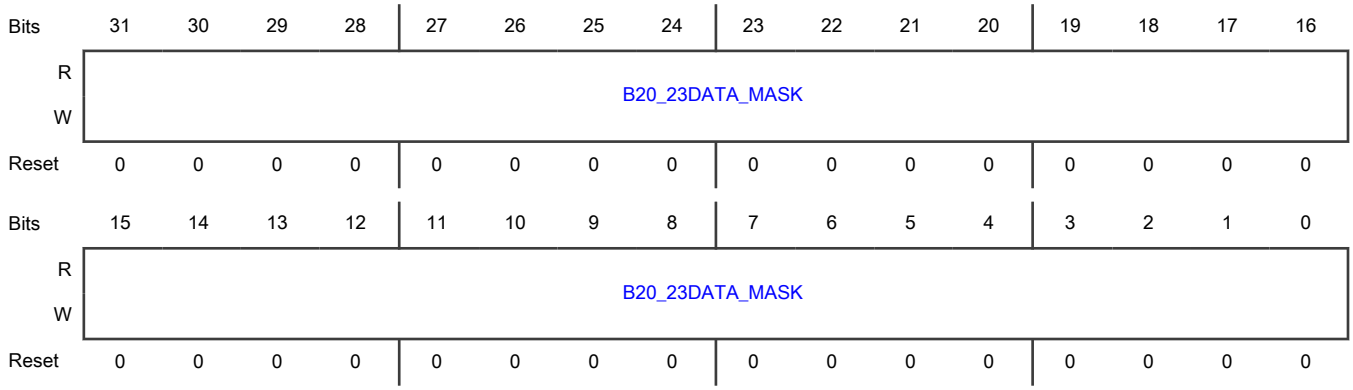
Offset

Register	Offset
EICHD19_WORD6	5D8h
EICHD20_WORD6	618h

Function

The seventh word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B20_23DATA_MASK correspond to bytes 20–23 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B20_23DATA_MASK	Data Mask Bytes 20-23 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B20_23DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 20-23 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 20-23 on the read data bus is inverted.

47.4.27 Error Injection Channel Descriptor 27, Word1 (EICHD27_WORD1)

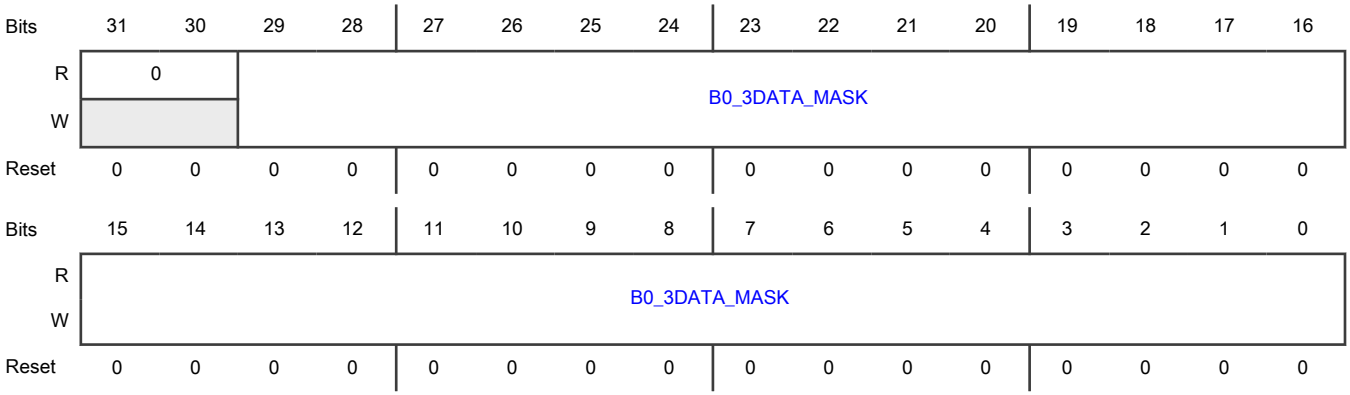
Offset

Register	Offset
EICHD27_WORD1	7C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-30 —	Reserved
29-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.4.28 Error Injection Channel Descriptor 28, Word1 (EICHD28_WORD1)

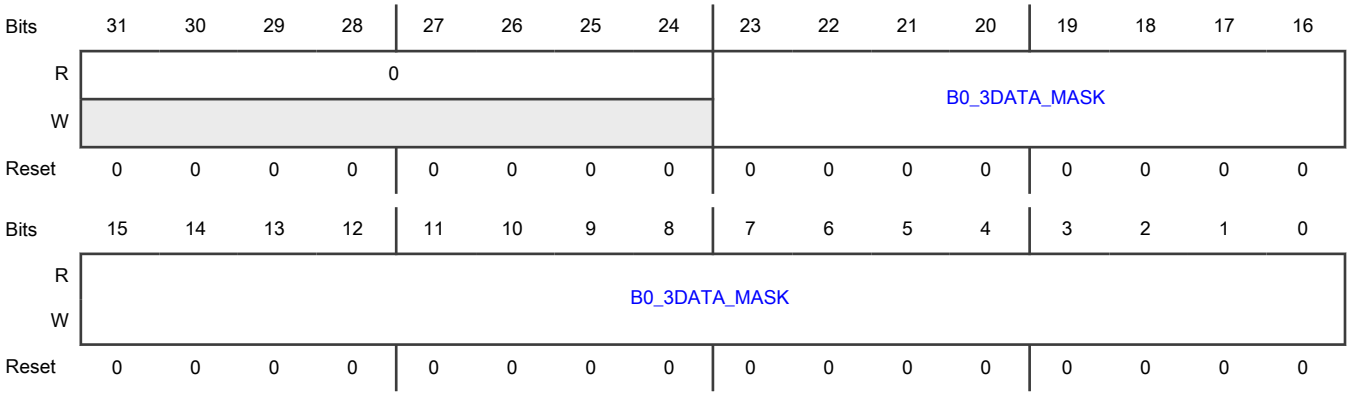
Offset

Register	Offset
EICHD28_WORD1	804h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.4.29 Error Injection Channel Descriptor n, Word1 (EICHD29_WORD1 - EICHD30_WORD1)

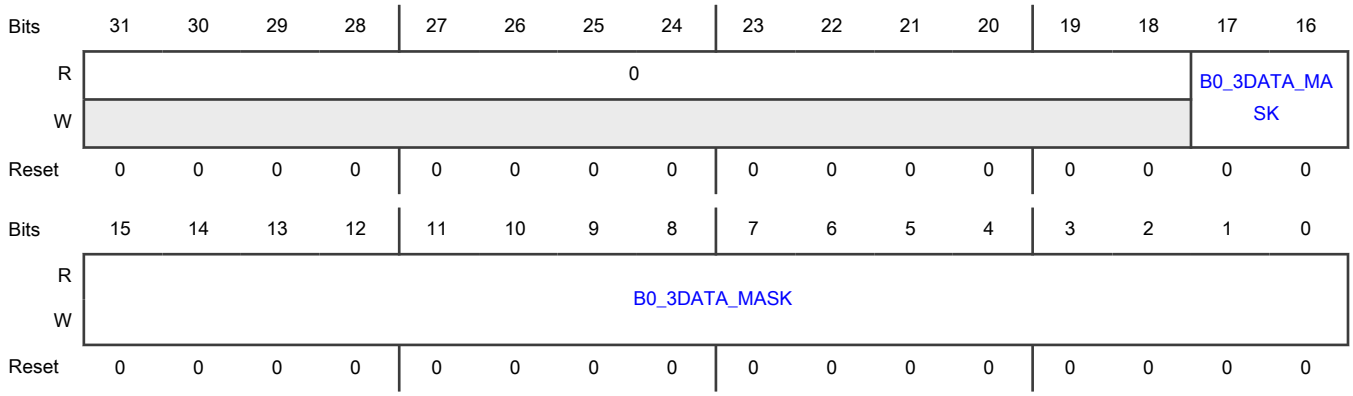
Offset

Register	Offset
EICHD29_WORD1	844h
EICHD30_WORD1	884h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-18 —	Reserved
17-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

47.5 Glossary

SEC-DED	Single error correct – dual error detect
ECC	Error correction code

Chapter 48
Error Reporting Module (ERM)

48.1 Chip-specific ERM information

48.1.1 ERM instances

This chip supports up to two instances of ERM:

- ERM_0
- ERM_1

Table 226. ERM instances

Instances	MWCT2D16S/MWCT2D17S/MWCT2016S/MWCT2015S
ERM_0	Yes
ERM_1	No

48.1.2 ERM channel mapping

ERM provides information on memory events associated with ECC and parity. It also provides you an option to enable interrupt notification for these events.

Table 227. ERM_0 channel mapping

Channel #	Module	Captured status
00	SRAM0	Single-bit error, multi-bit error, syndrome, absolute error address aligned to double-word (64-bit) boundary
01	SRAM1 [1]	Single-bit error, multi-bit error, syndrome, absolute error address+18000h aligned to double-word (64-bit) boundary[2]
02	Cortex-M7_0 I-cache tag RAM	Single-bit error, multi-bit error[3]
03	Cortex-M7_0 I-cache data RAM	Single-bit error, multi-bit error[3]
04	Cortex-M7_0 D-cache tag RAM	Single-bit error, multi-bit error[3]
05	Cortex-M7_0 D-cache data RAM	Single-bit error, multi-bit error[3]
06	Cortex-M7_1 I-cache tag RAM [4]	Single-bit error, multi-bit error[3]
07	Cortex-M7_1 I-cache data RAM	Single-bit error, multi-bit error[3]
08	Cortex-M7_1 D-cache tag RAM	Single-bit error, multi-bit error[3]
09	Cortex-M7_1 D-cache data RAM	Single-bit error, multi-bit error[3]
10	Cortex-M7_0 ITCM	Single-bit error, multi-bit error, syndrome, offset error address
11	Cortex-M7_0 D0TCM	Single-bit error, multi-bit error, syndrome, offset error address
12	Cortex-M7_0 D1TCM	Single-bit error, multi-bit error, syndrome, offset error address[5]
13	Cortex-M7_1 ITCM	Single-bit error, multi-bit error, syndrome, offset error address
14	Cortex-M7_1 D0TCM	Single-bit error, multi-bit error, syndrome, offset error address

Table continues on the next page...

Table 227. ERM_0 channel mapping (continued)

Channel #	Module	Captured status
15	Cortex-M7_1 D1TCM	Single-bit error, multi-bit error, syndrome, offset error address[5]
16	DMA TCD	Single-bit error, multi-bit error, syndrome, offset error address[6]
17	Flash memory port p0	Single-bit error, multi-bit error, absolute error address
18	Flash memory port p1	Single-bit error, multi-bit error, absolute error address
19	Flash memory port p2 [7]	Single-bit error, multi-bit error, absolute error address

1. SRAM1 is not available for the MWCT2D16S, MWCT2016S, and MWCT2015S product variants.
2. The size of SRAM0 is 160 KB, and therefore, to align addresses to the next power of 2, an address space of 256 KB is reserved for error reporting. However, SRAM0 and SRAM1 are in contiguous locations in the memory map. So, you must subtract 18000h (256 KB-160 KB = 96 KB that corresponds to 18000h) from the reported address to get an absolute address. See the "Memory and Memory Interfaces" chapter for SRAM details on different MWCT2xxxS variants.
3. The cache controller does not report error addresses and syndrome.
4. Cortex-M7_1 and its associated RAM and caches are not available in the MWCT2016S and MWCT2015S product variants.
5. For address reporting, bit 2 of the address is masked because this bit decides whether the access (read or write) is for D0TCM or D1TCM. For example, if the offset address is 0h or 8h, it is routed to D0TCM, but if the offset address is 4h or Ch, it is routed to D1TCM. The errors that are latched for these offset addresses are as follows:
   - Offset 0h : Address 0h is latched into the ERM channel corresponding to D0TCM.
   - Offset 4h : Address 0h is latched into the ERM channel corresponding to D1TCM.
   - Offset 8h : Address 8h is latched into the ERM channel corresponding to D0TCM.
   - Offset Ch : Address 8h is latched into the ERM channel corresponding to D1TCM.
6. Bits [31:10] and [2:0] are always 0 because they are not connected.

   Bits [9:5] indicate the corresponding TCD out of all the 32 implemented TCDs.

   Bits [4:3] indicate an offset of TCDs with respect to a 64-bit aligned boundary.
   - If [4:3] is 00, it indicates that an error is on offset address 20h.
   - If [4:3] is 01, it indicates that an error is on offset address 28h.
   - If [4:3] is 10, it indicates that an error is on offset address 30h.
   - If [4:3] is 11, it indicates that an error is on offset address 38h.
7. Flash memory port p2 is not applicable for the MWCT2016S and MWCT2015S product variants.

Table 228. ERM_1 channel mapping

Channel #	Module	Captured status
0	SRAM2	Single-bit error, multi-bit error, syndrome, absolute error address aligned to double, word (64-bit) boundary
1	-	-
2	CM7_2 I-cache tag RAM	Single-bit error, multi-bit error
3	CM7_2 I-cache data RAM	Single-bit error, multi-bit error
4	CM7_2 D-cache tag RAM	Single-bit error, multi-bit error
5	CM7_2 D-cache data RAM	Single-bit error, multi-bit error
6	CM7_3 I-cache tag RAM	Single-bit error, multi-bit error
7	CM7_3 I-cache data RAM	Single-bit error, multi-bit error

Table continues on the next page...

Table 228. ERM_1 channel mapping (continued)

Channel #	Module	Captured status
8	CM7_3 D-cache tag RAM	Single-bit error, multi-bit error
9	CM7_3 D-cache data RAM	Single-bit error, multi-bit error
10	CM7_2 ITCM	Single-bit error, multi-bit error, syndrome, offset error address
11	CM7_2 D0TCM	Single-bit error, multi-bit error, syndrome, offset error address
12	CM7_2 D1TCM	Single-bit error, multi-bit error, syndrome, offset error address
13	CM7_3 ITCM	Single-bit error, multi-bit error, syndrome, offset error address
14	CM7_3 D0TCM	Single-bit error, multi-bit error, syndrome, offset error address
15	CM7_3 D1TCM	Single-bit error, multi-bit error, syndrome, offset error address
16	-	-
17	Flash memory port p3	Single-bit error, multi-bit error, syndrome, offset error address
18	-	-
19	-	-
20	-	-
21	-	-
22	-	-
23	-	-

48.2 Overview

The Error Reporting Module (ERM) provides information and optional interrupt notification on memory error events associated with ECC and parity. The ERM collects error events on memory accesses for memory arrays, such as flash memory, system RAM, or peripheral RAMs. ERM supports various channels for memory sources where each ERM channel is associated with a different memory module. See the chip-specific ERM information for details about supported memory sources and specific memory channel assignments. If memory supports ECC then ERM syndrome and error address information is captured along with error event. ERM does not receive this information in case of cache or memory with parity along with error event.

48.2.1 Features

The ERM includes these features:

- Optional interrupt notification on captured error events
- Capturing of address and syndrome information on single-bit correction and non-correctable ECC events
- Support for error event capturing for memory sources, with individual reporting fields and interrupt configuration per memory channel
- Recording the count value of the number of corrected error events

48.3 Functional description

48.3.1 Single-bit correction events

When a single-bit correction event on Memory *n* is detected, the ERM:

- Records the event by changing the value of the applicable Status Register bit SR*x*[SBC*n*] to 1.

- Increments the correctable error count value (until the counter reaches its maximum value): CORR_ERR_CNT*n*[COUNT].
- Records the corresponding access address that initiated the event in the Memory *n* Error Address Register: EAR*n* (if this register is present for the channel).
- Stores the corresponding ECC syndrome in the Memory *n* Error Syndrome Register: SYN*n* (if this register is present for the channel). This register identifies the bit position of the corrected data on single-bit data inversion.

The ERM holds event information only for the last reported event.

To clear the record of an event, write 1 to SR*x*[SBC*n*] to change its value to 0.

To reset the correctable error count value, write all zeros to CORR_ERR_CNT*n*[COUNT].

Optional interrupt notification for single-bit correction events

The ERM provides an option to generate an interrupt notification upon the report of a single-bit correction event. This sequence describes how to use the option:

1. To enable interrupt notification for single-bit correction events on Memory *n*, set CR*x*[ESCIE*n*] to 1.
2. Subsequently, when a single-bit correction event on Memory *n* is detected, the ERM:
   - Records the event and address, and stores the ECC syndrome as usual.
   - Additionally sends an interrupt notification corresponding to the event.
3. To clear both the record of an event and the corresponding interrupt notification, write 1 to SR*x*[SBC*n*] to change its value to 0.

48.3.2 Non-correctable error events

When a non-correctable ECC error event on Memory *n* is detected, the ERM:

- Records the event by changing the value of the applicable Status Register bit: SR*x*[NCE*n*] to 1.
- Records the corresponding access address that initiated the event in the Memory *n* Error Address Register: EAR*n* (if this register is present for the channel).
- Stores the corresponding ECC syndrome in the Memory *n* Error Syndrome Register: SYN*n* (if this register is present for the channel).
  - In the event of a non-correctable address bit inversion, SYN*n* identifies the pertinent address bit position.
  - In the event of a non-correctable, multi-bit data inversion, the syndrome value does not provide any additional diagnostic information.

The ERM holds event information only for the last reported event.

To clear the record of an event, write 1 to SR*x*[NCE*n*] to change its value to 0.

Optional interrupt notification for non-correctable error events

The ERM provides an option to generate an interrupt notification upon the report of a non-correctable ECC event. This sequence describes how to use the option:

1. To enable interrupt notifications for non-correctable error events on Memory *n*, set CR*x*[ENCIE*n*] to 1.
2. Subsequently, when a non-correctable error event on Memory *n* is detected, the ERM:
   - Records the event and address and stores the ECC syndrome as usual.
   - Additionally sends an interrupt notification corresponding to the event.
3. To clear both the record of an event and the corresponding interrupt notification, write 1 to SR*x*[NCE*n*] to change its value to 0.

48.4 Initialization

For each ERM channel supporting memory with ECC, prepare the corresponding memory array before enabling ERM interrupts about errors for that memory.

1. Initialize the memory to a known value so that the correct corresponding ECC codeword is stored.
2. During the memory's initialization, if the ERM captures information about any ECC error event, clear the corresponding SR*x*[SBC*n*] or SR*x*[NCE*n*] field that stores the record of the event.
3. Program the applicable CR*x*[ESCIE*n*] and CR*x*[ENCIE*n*] fields to enable ERM interrupts as desired.

48.5 ERM register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

48.5.1 ERM memory map

ERM base address: 4025_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	RW	0000_0000h
14h	ERM Status Register 1 (SR1)	32	RW	0000_0000h
18h	ERM Status Register 2 (SR2)	32	RW	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	R	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	R	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)	32	RW	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	R	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	R	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT2)	32	RW	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT3)	32	RW	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT4)	32	RW	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT5)	32	RW	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT6)	32	RW	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT7)	32	RW	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT8)	32	RW	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT9)	32	RW	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	R	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	R	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_CNT10)	32	RW	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	R	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	R	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_CNT11)	32	RW	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	R	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	R	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_CNT12)	32	RW	0000_0000h
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	R	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	R	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	RW	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	R	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	RW	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	R	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	R	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	RW	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	R	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	R	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_CNT16)	32	RW	0000_0000h
210h	ERM Memory 17 Error Address Register (EAR17)	32	R	0000_0000h
218h	ERM Memory 17 Correctable Error Count Register (CORR_ERR_CNT17)	32	RW	0000_0000h
220h	ERM Memory 18 Error Address Register (EAR18)	32	R	0000_0000h
228h	ERM Memory 18 Correctable Error Count Register (CORR_ERR_CNT18)	32	RW	0000_0000h
230h	ERM Memory 19 Error Address Register (EAR19)	32	R	0000_0000h
238h	ERM Memory 19 Correctable Error Count Register (CORR_ERR_CNT19)	32	RW	0000_0000h

48.5.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

48.5.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE8	ENCIE8	0		ESCIE9	ENCIE9	0		ESCIE10	ENCIE10	0		ESCIE11	ENCIE11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE12	ENCIE12	0		ESCIE13	ENCIE13	0		ESCIE14	ENCIE14	0		ESCIE15	ENCIE15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 non-correctable error events is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

48.5.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

Function

This 32-bit control register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE16	ENCIE16	0		ESCIE17	ENCIE17	0		ESCIE18	ENCIE18	0		ESCIE19	ENCIE19	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE17	ESCIE17 Enable Memory 17 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 17 single-bit correction events is disabled. 1b - Interrupt notification of Memory 17 single-bit correction events is enabled.
26 ENCIE17	ENCIE17 Enable Memory 17 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 17 non-correctable error events is disabled. 1b - Interrupt notification of Memory 17 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE18	ESCIE18 Enable Memory 18 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 18 single-bit correction events is disabled. 1b - Interrupt notification of Memory 18 single-bit correction events is enabled.
22 ENCIE18	ENCIE18 Enable Memory 18 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 18 non-correctable error events is disabled. 1b - Interrupt notification of Memory 18 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE19	ESCIE19 Enable Memory 19 Single Correction Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 19 single-bit correction events is disabled. 1b - Interrupt notification of Memory 19 single-bit correction events is enabled.
18 ENCIE19	ENCIE19 Enable Memory 19 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 19 non-correctable error events is disabled. 1b - Interrupt notification of Memory 19 non-correctable error events is enabled.
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

48.5.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3	SBC7

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC7	Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

48.5.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23	SBC10

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC10	Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

48.5.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		SBC17	NCE17	0		SBC18	NCE18	0		SBC19	NCE19	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-28 —	Reserved
27 SBC17	SBC17 Memory 17 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE17] is enabled. 0b - No single-bit correction event on Memory 17 detected. 1b - Single-bit correction event on Memory 17 detected.
26 NCE17	NCE17 Memory 17 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE17] is enabled. 0b - No non-correctable error event on Memory 17 detected. 1b - Non-correctable error event on Memory 17 detected.
25-24 —	Reserved
23 SBC18	SBC18 Memory 18 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE18] is enabled. 0b - No single-bit correction event on Memory 18 detected. 1b - Single-bit correction event on Memory 18 detected.
22 NCE18	NCE18 Memory 18 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE18] is enabled. 0b - No non-correctable error event on Memory 18 detected. 1b - Non-correctable error event on Memory 18 detected.
21-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19 SBC19	SBC19 Memory 19 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE19] is enabled. 0b - No single-bit correction event on Memory 19 detected. 1b - Single-bit correction event on Memory 19 detected.
18 NCE19	NCE19 Memory 19 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE19] is enabled. 0b - No non-correctable error event on Memory 19 detected. 1b - Non-correctable error event on Memory 19 detected.
17-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

48.5.8 ERM Memory a Error Address Register (EAR0 - EAR19)

Offset

Register	Offset
EAR0	100h

Table continues on the next page...

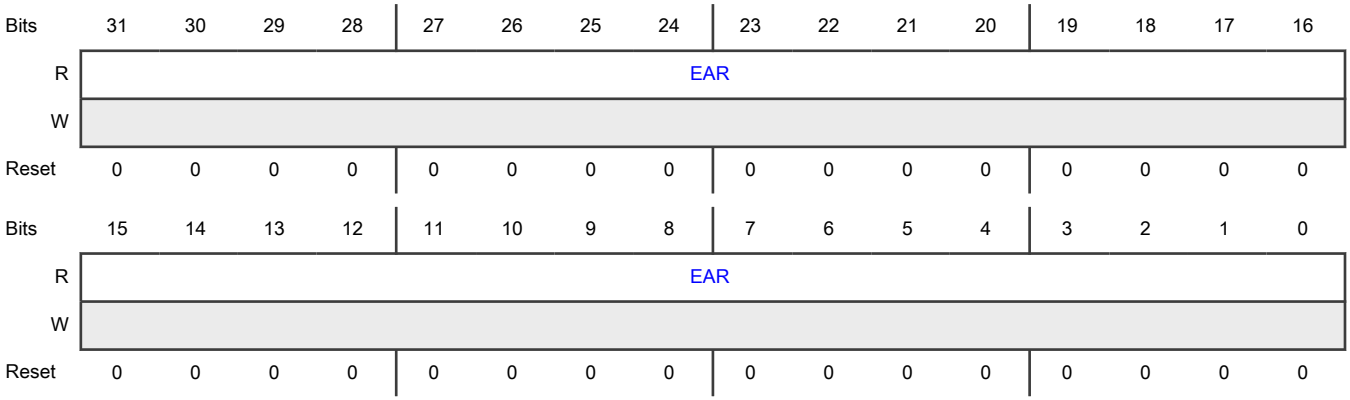
Table continued from the previous page...

Register	Offset
EAR1	110h
EAR10	1A0h
EAR11	1B0h
EAR12	1C0h
EAR13	1D0h
EAR14	1E0h
EAR15	1F0h
EAR16	200h
EAR17	210h
EAR18	220h
EAR19	230h

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

48.5.9 ERM Memory a Syndrome Register (SYN0 - SYN16)

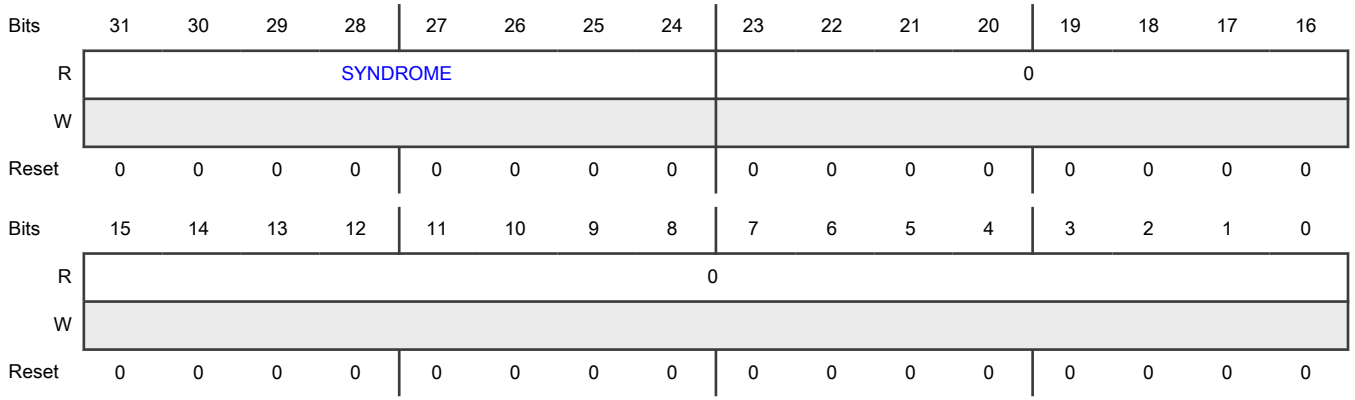
Offset

Register	Offset
SYN0	104h
SYN1	114h
SYN10	1A4h
SYN11	1B4h
SYN12	1C4h
SYN13	1D4h
SYN14	1E4h
SYN15	1F4h
SYN16	204h

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

48.5.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT19)

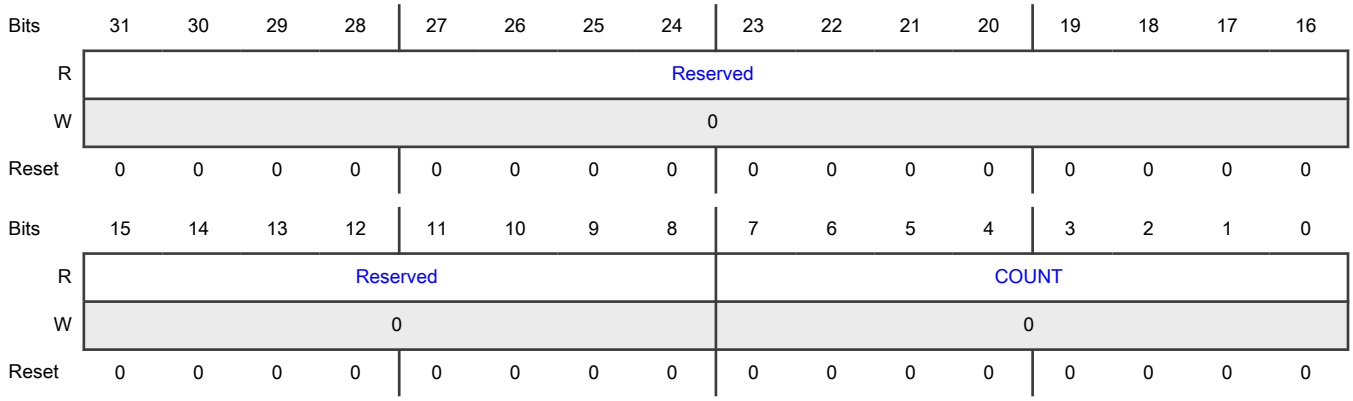
Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h
CORR_ERR_CNT17	218h
CORR_ERR_CNT18	228h
CORR_ERR_CNT19	238h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

48.6 Glossary

ECC Error correction code

Chapter 49
Fault Collection and Control Unit (FCCU)

49.1 Chip-specific FCCU information

49.1.1 FCCU NCF slots

Table 229. FCCU NCF slots

Slot number	Source module (error type)
NCF[0]	• Cortex-M7 LS and core lockup • HSE_B lockup
NCF[1]	Interconnect: • All EDC bus gaskets • XBIC monitors and platform gaskets
NCF[2]	ECC errors: • PRAMC • TCMs • Caches • eDMA • EDC after ECC • HSE_B RAM errors
NCF[3]	All flash memory errors: • FMU • PFLASH • DCM flash memory
NCF[4]	Voltage-related errors: • PMC 1.1 V and 2.5 V GNG • Pad overvoltage
NCF[5]	Debug and test monitoring
NCF[6]	INTM
NCF[7]	Software notification

49.1.2 Chip-boundary FCCU signals

Table 230. Chip-boundary FCCU signals

FCCU signal	Chip signal
EOUT0	FCCU_ERR0
EOUT1	FCCU_ERR1

49.1.3 FCCU clocking

Table 231. FCCU clocking

FCCU clock signal	Chip clock signal
CLKSAFE	FIRC_CLK
CLKPRIM	AIPS_PLAT_CLK

49.1.4 FOSU timer interval

The FOSU_COUNT value determines the FOSU timer interval. On this chip, FOSU_COUNT is 69780h.

49.1.5 Supported internal chip reactions

The short functional reset discussed in this chapter is equivalent to the chip functional reset. See the interrupt map file attached to this document for interrupts from FCCU to NVIC.

NOTE

After STCU2 completes the self-testing procedure, the chip reboots and FCCU resets.

49.1.6 Supported FCCU EOUT fault output modes (protocols)

This chip supports only the Bistable protocol of FCCU EOUT, and does not support the other protocols discussed in this chapter, Dual-Rail and Time-Switching.

49.1.7 Recommended reaction programming for faults

You can upgrade or downgrade the reaction of the faults according to the recommended reaction discussed in the fault map file attached to this document.

In case you upgrade a reaction, no issues are expected in the behavior. If you downgrade the reaction, the functionality is not guaranteed.

This is the recommendation for faults caused by lockstep errors:

1. The recommended reaction for the lockstep error fault is a functional reset.
2. Program the core to perform the following steps after rebooting:
    a. Initialize TCM (because TCMs can become corrupted)
    b. Invalidate the cache (because caches can become corrupted)
3. Write 1 to the corresponding bit in FCCU.NCF_S0 register to clear the non-critical fault status.
4. If the FCCU fault gets cleared, you have nothing else to do for fault handling (this means, the FCCU lockstep error is recovered).
5. If the error persists, it indicates that the internal registers of the two cores have different values. To bring them to the same value, you could perform any of these steps:
    a. Initialize destructive reset through software by using MC_ME.MODE_CONF[DEST_RST] to recover from lockstep. This initializes the FCCU with no pending faults.
    b. Reconfigure the Cortex-M7 debug configurations (which would have been lost in previous step due to destructive reset in the system).

49.1.8 FCCU NCF handling architecture

This chip supports eight NCFs. Therefore, multiple faults of similar nature are ORed together and then connected as a single NCF to FCCU. To control individual fault, there are enable/disable controls provided within DCM in registers DCMRWD3, DCMRWD4 and DCMRWD5.

Similarly, the FCCU.NCF_S0 captures the status of fault within FCCU. This fault might have resulted due to any fault mapped on the corresponding FCCU NCF channel. The DCM status registers, DCMROD3, DCMROD4 and DCMROD5 capture the status of individual faults which are mapped onto FCCU NCFs. Figure 183 shows this arrangement.

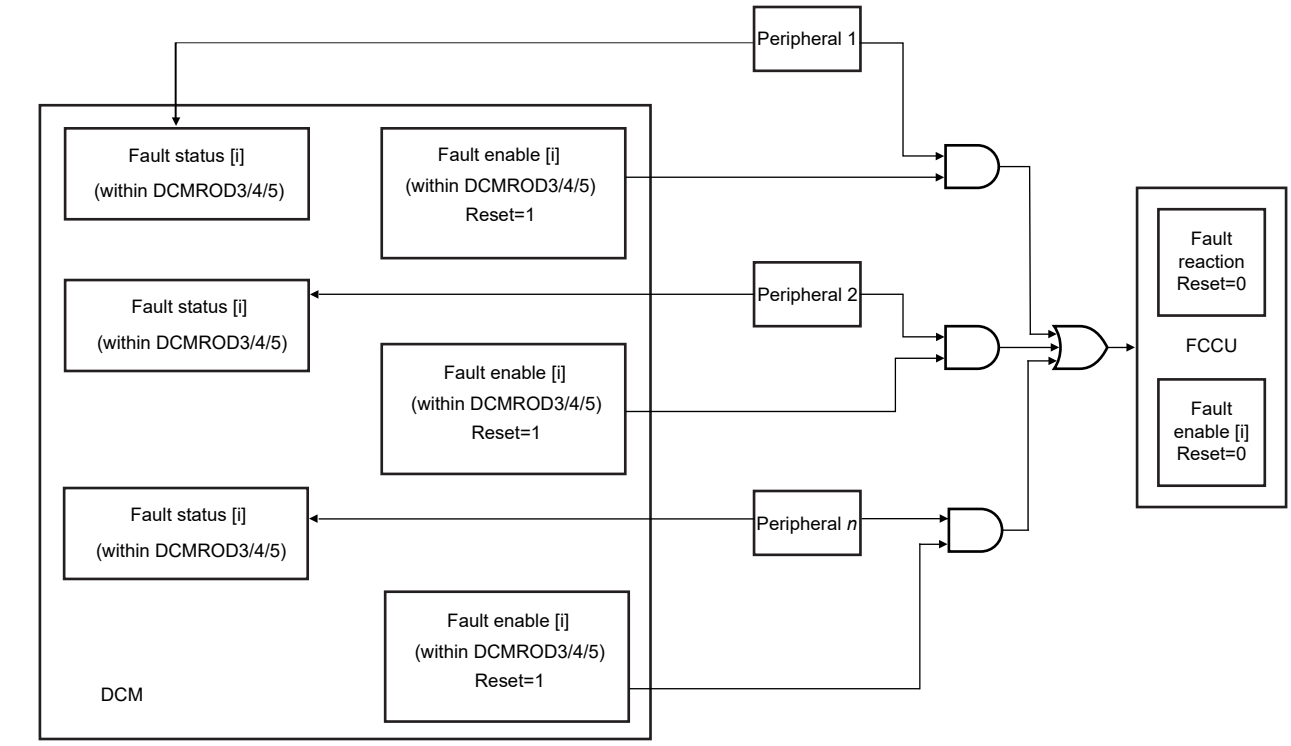


Figure 183. FCCU NCF handling architecture

49.2 Introduction

The FCCU provides a hardware interface to collect faults and to place the device into a safe state when a failure is detected in the device. No CPU intervention is requested for collection and control operations. FCCU offers a systematic approach to fault collection and control.

49.3 Features

The key features of the FCCU module are these:

- Management of non-critical faults
- HW or SW fault recovery management
- Fault collection from safety relevant modules on the device
- Fault injection (fake faults)
- Collection of test results
- Lockable configuration
  - Changes are only possible after entering the CONFIG state

  - Supports a transient and a permanent lock
  - Configuration changes observed by a watchdog timer
- Configurable fault control
- External reaction (FAULT state): EOUT signaling. Error indication via the pin(s) is controlled by FCCU.
- Internal chip reactions (ALARM state): interrupt request
- Configurable internal chip reactions for each NCF (FAULT state):
  - Short functional reset request pulse
  - NMI
  - No reaction
  - IRQ
- In Bi-Stable operational mode, one of the EOUT signals is high to indicate an OK operational state of FCCU.
- After power on, the EOUT signals have high impedance.[6] They indicate an operational state only after the software configures them.
- In case of a failure event or on software request for error pin indication, the pin(s) are set to faulty state for a minimum time T_{min} (see DELTA_T[DELTA_T]), even if the software tries to release it before (for the case of error pin configured in Bi-Stable mode only).

The self-checking procedure checks the FCCU circuitry at the start up. The FCCU is operational with the default configuration immediately after the completion of the self-checking procedure. Internal (short functional reset request pulse, interrupt request) and external (EOUT signaling) reactions are statically defined or programmable. The default configuration can be modified only in the Configuration (CONFIG) state. FCCU is designed to function when CLKPRIM is faster than the CLKSAFE clocks.

49.4 Block diagram

The following figure represents a top-level diagram of the FCCU module.

[6] Actual value depends on device-specific setting at pad level.

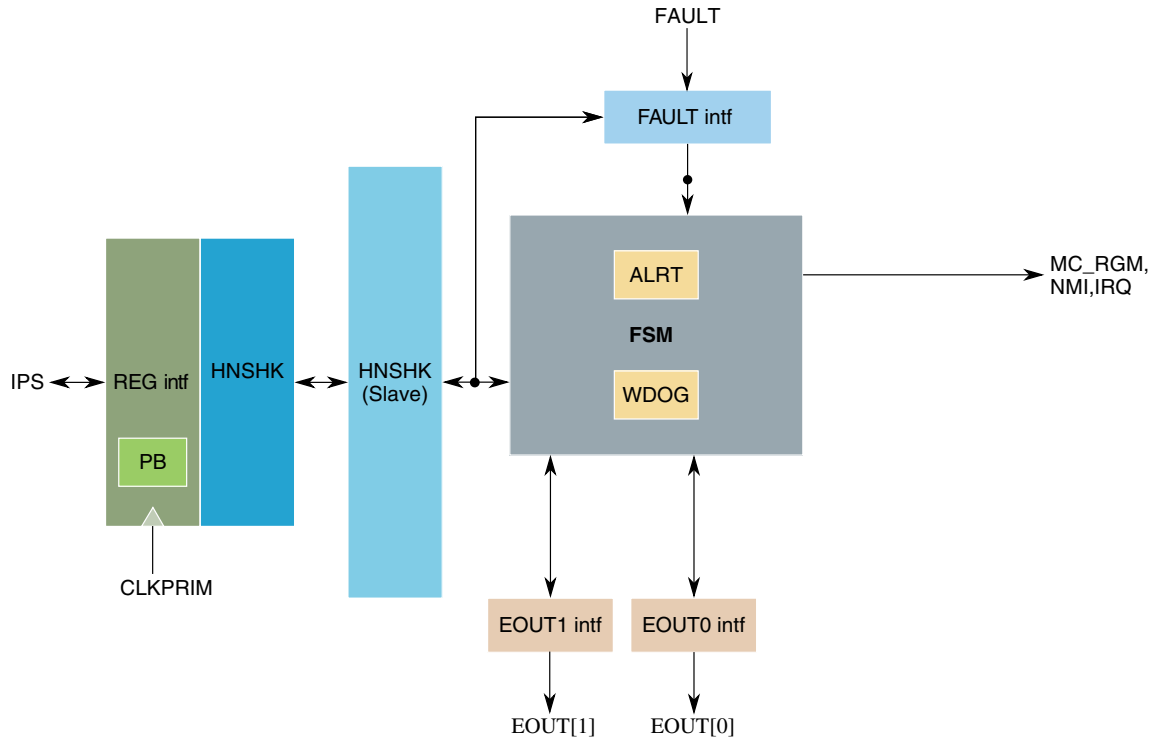


Figure 184. FCCU block diagram

This table describes the FCCU submodules.

Table 232. FCCU submodules

Submodule	Description
REG intf	Includes the register file, the IPS bus interface, the IRQ interface and the parity block (PB) for the configuration registers
HNSHK blocks (master and slave blocks)	Includes the FSM ability to support the handshake between the REG interface and the FSM unit because of the usage of two asynchronous clocks [CLKPRIM(module clock) and CLKSAFE(RC oscillator clock)]
FSM unit	Implements the main functions of FCCU. The FSM also includes the: • Watchdog timer (WDG) • Alarm timer (ALRT)
FAULT intf	Implements the interface for fault conditioning and management
EOUTx units	Implement the output stage to manage the EOUT interfaces

49.5 Signal description

49.5.1 Signals

This table describes the signals on the boundary of FCCU.

Signal	Direction	Function	Sensitivity	Clock
Clock and reset				
CLKPRIM	Input	Primary Clock—supplies the primary clock.	Rising edge	—
CLKSAFE0	Input	Safe Clock 0—supplies one of two 48 MHz clocks for use by various redundant submodules. This clock and CLKSAFE1 are often referred to as the same clock (CLKSAFE).	Rising edge	—
CLKSAFE1	Input	Safe Clock 1—supplies one of two 48 MHz clocks for use by various redundant submodules. This clock and CLKSAFE0 are often referred to as the same clock (CLKSAFE).	Rising edge	—
Fault interface				
NCF[7:0]	Input	Non-critical Fault Channels—receive indications of non-critical faults from various sources. See the fault map file attached to this document.	Programmable	Asynchronous
EOUT interface				
EIN[1:0]	Input	Error Inputs—provide the mechanism for FCCU to capture the states that off-chip logic drives on the associated EOUT[1:0] signals. FCCU captures the states in the EINOUT[EIN1] and EINOUT[EIN0] fields.	—	Asynchronous
EOUT[1:0]	Output	Error Outputs—indicate FCCU's condition (faulty, nonfaulty, or configuration) to off-chip logic.	Programmable	CLKSAFE0, CLKSAFE1
FOSU interface				
FIF	Output	FCCU In Fault—indicates to the FOSU module, if present, that FCCU is recognizing a fault. FCCU asserts and then deasserts this signal in two cases: *Case 1* When the EOUT signals are programmed to be always low (CFG[FCCU_SET_CLEAR]), FCCU	High	CLKSAFE0, CLKSAFE1

Table continues on the next page...

Table continued from the previous page...

Signal	Direction	Function	Sensitivity	Clock
		asserts this signal until the EOUT signals are programmed not to be always low. *Case 2* When the EOUT signals are programmed not to be always low (CFG[FCCU_SET_CLEAR]), FCCU asserts this signal when all of the following are true: • FCCU enters the FAULT state as the result of a fault. • If the fault is a non-critical fault, the associated EOUT_SIG_ENa[EOUTENn] field is set to enabled. FCCU deasserts this signal when FCCU leaves the FAULT state.		
Interrupt interface				
NMIOUT	Output	Non-maskable Interrupt Output—Sends a non-maskable interrupt request to the processor core or cores when FCCU enters the FAULT state as the result of a non-critical fault for which NMI is enabled as the reaction (NMI_ENa[NMIENn]).	Low	CLKPRIM

49.6 Functional description

49.6.1 Definitions

In general, the following definitions are applicable for fault management:

- HW recoverable fault: The fault indication is a level-sensitive signal that remains asserted until the fault cause is deasserted. That is, if logical 0 on the fault signal indicates fault, then the status flags are valid as long as the fault line stays at 0. The status is automatically cleared when the fault signal goes to 1. Typically the fault signal is latched external to the FCCU in the module where the fault occurred. The FCCU state transitions are consequently executed on the state changes of the input fault signal. No SW intervention in the FCCU is required to recover the fault condition.
- SW recoverable fault: The fault indication is a signal asserted without a defined time duration. The fault signal is latched in the FCCU. The fault recovery is executed following a SW recovery procedure (status/flag register clearing).

HW recoverable is an option to exclude the handling of error source/s by FCCU management SW, in case it is known that the fault is recoverable by itself when the fault condition is corrected.

These type of chip resets reset the FCCU:

- Power-on
- Destructive

A short functional reset of the chip do not reset the FCCU.

For more information on each type of reset, see the Reset Generation Module (MC_RGM) and reset chapters.

49.6.2 FSM description

The functionality of FCCU is depicted by the FSM state diagram (see Figure 185).

FCCU has four states that are identified with the following meaning:

- CONFIG: Used only to modify the configuration of FCCU from its default. A subset of the FCCU registers, dedicated to define the FCCU configuration (global configuration, reactions to fault, timeout, non-critical fault masking) can be accessed in write mode only in the CONFIG state.

  The CONFIG state is accessible only in the NORMAL state and if the configuration is not locked. A permanent configuration lock can be disabled by a reset that also resets the FCCU. The transient lock register is unlocked by writing BCh into it. FCCU gets transiently locked again if an invalid key is written into TRANS_LOCK[TRANSKEY] (that is, other than BCh). To lock FCCU for configuration, write FFh to PERMNT_LOCK[PERMNTKEY].

  After the release of reset, the state of the transient lock is locked, and the state of the permanent lock is unlocked.

  The locking feature only restricts the FSM movement into CONFIG state. After the user enters the CONFIG state and then tries to lock the configuration, the locking of configuration is effective only after FCCU moves to the NORMAL state; it will not be effective in the current CONFIG state.

  The CONFIG to NORMAL state transition can be executed by SW or automatically following a timeout condition of the watchdog. In case the timeout information and the SW request for state change to NORMAL appears at the same time, watchdog timeout has the priority and hence the configuration registers (those that are writable only in the CONFIG state) are reset to their default values. The movement to the NORMAL state is made.

  The incoming faults, occurring during the configuration phase (CONFIG state) are latched in order to process them when FCCU is moved to the NORMAL state, according to the new configuration.

  All pending faults that occur during the CONFIG state result in both of the following:

  — Highest-priority state transition

  — Interrupt generation (NMI or alarm IRQ)

  If the state transition occurs, it gives the reset reaction corresponding to the worst case based on all the faults (pending or non-pending faults) that occurred during the CONFIG state.

- NORMAL: This is FCCU's operating state when no faults are occurring. It is also the default state on the reset exit. Following state transitions occur on one of the following events:

  — Unmasked non-critical faults with the timeout disabled: FCCU moves to the FAULT state.

  — Unmasked non-critical faults with the timeout enabled: FCCU moves to the ALARM state.

  — Masked non-critical faults: FCCU stays in the NORMAL state.

- ALARM: FCCU moves into the ALARM state when an unmasked non-critical fault occurs and the timeout is enabled. Transition to the ALARM state goes along with an interrupt alarm, if enabled. By definition, this fault may be recovered within a programmable timeout period, before it generates a transition to the FAULT state. The timeout is reinitialized if FCCU enters the NORMAL state. The timeout restarts following the recovery from the FAULT state.

- FAULT: FCCU moves into the FAULT state when one of the following condition occurs:

  — Timeout related to a non-critical fault when FCCU is in the ALARM state

  — Unmasked non-critical faults with the timeout disabled

The transition from the NORMAL or ALARM to the FAULT state goes along with the generation of:

- Internal chip reaction—NMI interrupt (optional)
- External reaction—EOUT signaling (optional)

- Internal chip reaction—SW option: Soft reaction (Short functional reset request pulse if configured)
- Non Maskable Interrupt (NMI) is routed to all cores.

After moving to the FAULT state, if there is either a previous pending fault or a new fault for which NMI is enabled, NMI generation takes place.

Multiple faults can occur at the same time.

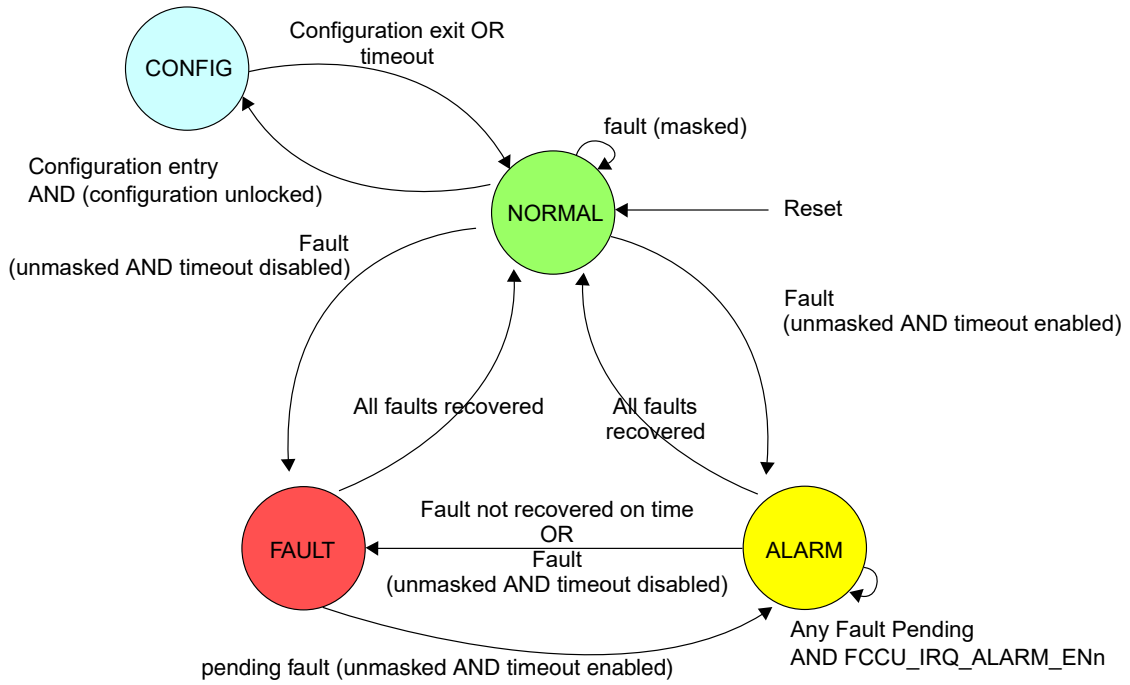


Figure 185. FCCU state diagram

49.6.3 Fault priority scheme and nesting

The FAULT state has a higher priority than the ALARM state in case of concurrent fault events (non-critical) that occur in the NORMAL state.

The ALARM to FAULT state transition occurs if a non-critical fault (unmasked and with timeout disabled) is asserted in the ALARM state.

The ALARM to NORMAL state transition occurs only if all the non-critical faults (including the faults that have been collected after the entry in the ALARM state) have been cleared (SW or HW recovery); otherwise the FCCU remains in the ALARM state.

The FAULT to NORMAL state transition occurs only if all the non-critical faults (including the faults that have been collected after the entry in the FAULT/ALARM state) have been cleared (SW or HW recovery); otherwise the FCCU moves to the ALARM state (if any non-critical fault is still pending and the timeout is not elapsed).

In general, no fault nesting is supported except for the non-critical faults that cause an ALARM to FAULT state transition. In this case, the NCF timer is stopped until the FAULT state is recovered. If FCCU is in the ALARM state and another fault occurs, which has its alarm timeout enabled, then the alarm timer shall not reload and shall not start again.

49.6.4 Fault recovery

The following timing diagrams describe the main use cases of FCCU in terms of fault events and related recovery.

A typical sequence related to non-critical fault management (ALARM state), see Figure 186 and Figure 187, is as follows:

1. Non-critical fault assertion

2. FCCU state transition (automatic): NORMAL to ALARM
   - Alarm interrupt request (if enabled)
   - Timeout running
3. System state: RUN
4. Alarm interrupt management: fault recovery
   - FCCU state transition: ALARM to NORMAL

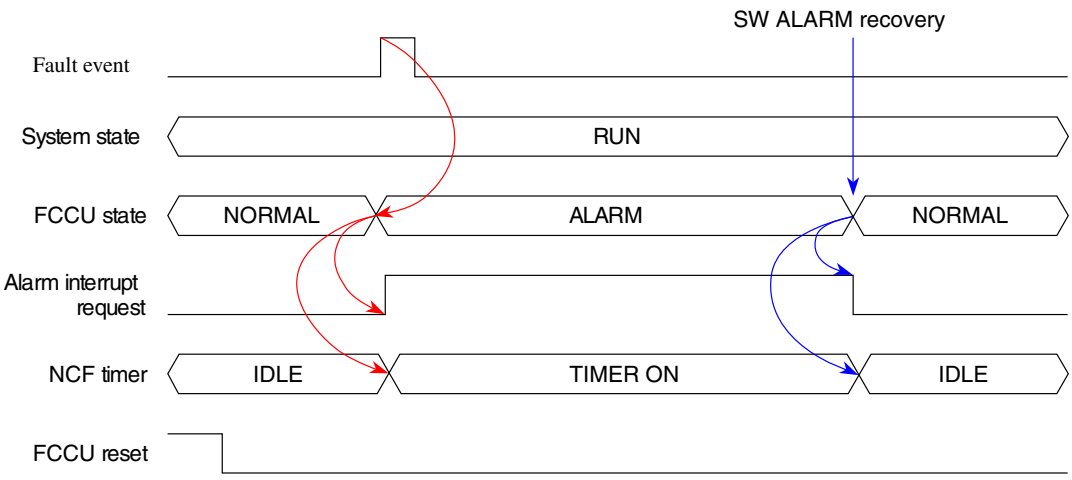


Figure 186. Non-critical fault (ALARM state) SW recovery

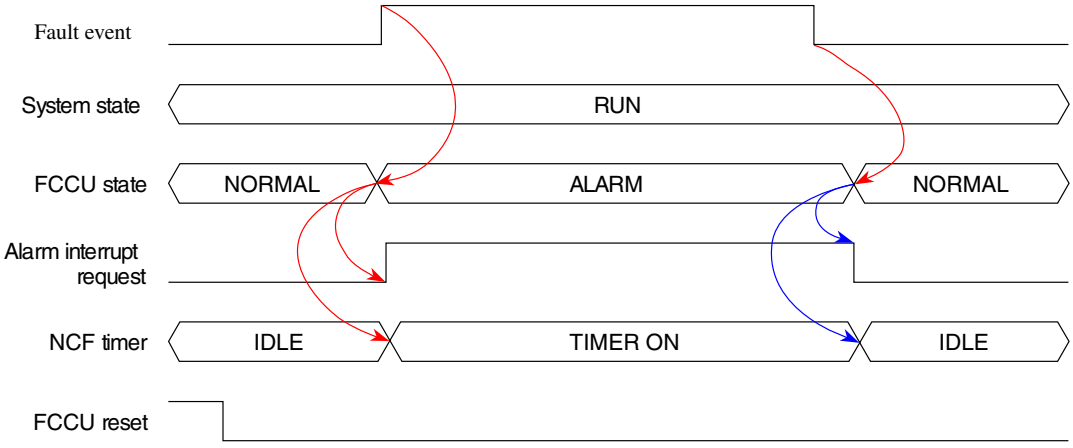


Figure 187. Non-critical fault (ALARM state) HW recovery

A typical sequence related to non-critical fault management (ALARM to FAULT state), see Figure 188, is as follows:

1. Non-critical fault assertion
2. FCCU state transition (automatic): NORMAL to ALARM
   - Alarm interrupt request (if enabled)
   - Timeout running
3. FCCU state transition (following the timeout trigger): ALARM to FAULT
   - NMI assertion (if enabled)
4. NMI interrupt management (if enabled)

- Fault recovery (by software): FCCU state transition: FAULT to NORMAL

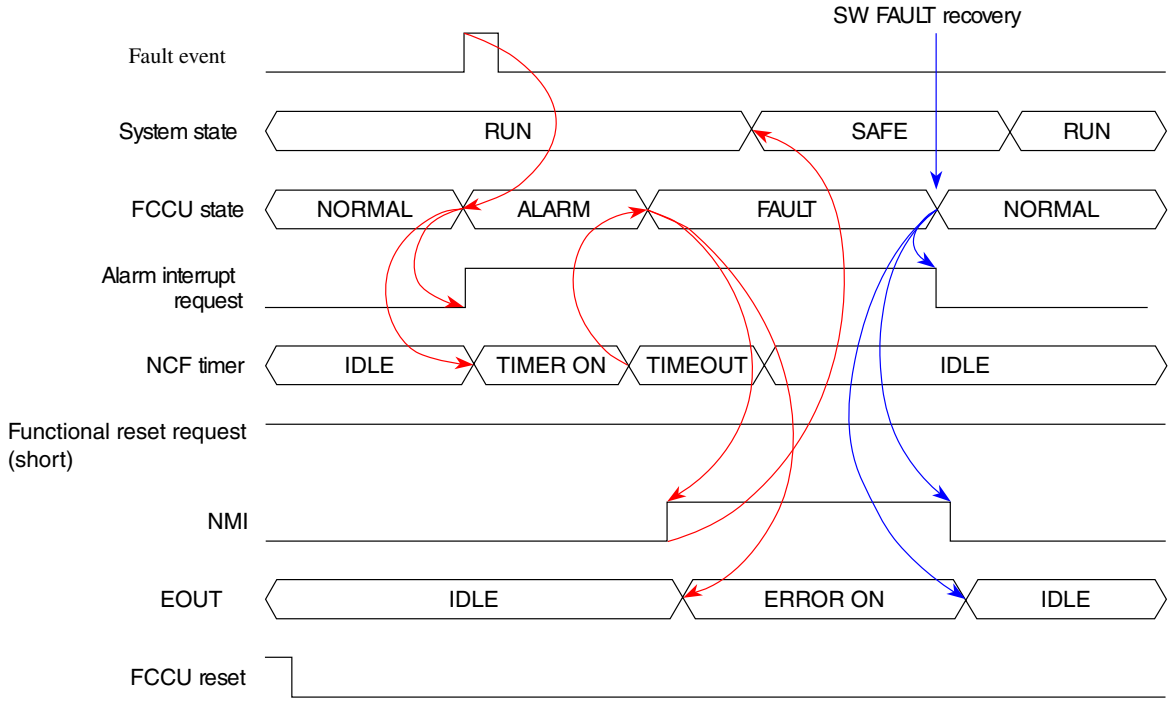


Figure 188. Non-critical fault (ALARM to FAULT state) recovery

49.6.5 EOUT interface

Introduction

You use the EOUT[1:0] signals to indicate FCCU's condition to off-chip logic.

NOTE

For information on the availability and names of these FCCU signals on the boundary of this chip, see the chip-specific FCCU information.

The FCCU conditions

There are three FCCU conditions:

Condition	Description
Faulty	All of the following are true: • The fault-output (EOUT) timer is running (see How the fault-output (EOUT) timer works in Bi-Stable fault-output mode). • FCCU is in FAULT state.
Nonfaulty	All of the following are true:

Table continues on the next page...

Table continued from the previous page...

Condition	Description
	• The fault-output (EOUT) timer is not running. • FCCU is in ALARM or NORMAL state.
Configuration	All of the following are true: • The fault-output (EOUT) timer is not running. • FCCU is in CONFIG state.

How the fault-output (EOUT) timer works in Bi-Stable fault-output mode

In Bi-Stable fault-output mode (FOM), FCCU starts the fault-output (EOUT) timer when all of the following are true:

- If the EOUT signals are in Bi-Stable FOM, and the EOUT signals are not programmed to be always low (CFG[FCCU_SET_CLEAR]).
- The EOUT timer is not already running.
- FCCU enters the FAULT state as the result of a fault.

When the fault-output (EOUT) timer is already running and a new fault occurs:

- If FCCU is in the CONFIG state: FCCU does not restart the EOUT timer.
- If FCCU is in the NORMAL or ALARM state:
  - And ALARM state is enabled for the fault (non-critical): FCCU enters (or remains in) the ALARM state but does not restart the EOUT timer.
  - And ALARM state is disabled for the fault (non-critical): FCCU enters the FAULT state and restarts the EOUT timer.
- If FCCU is in the FAULT state: FCCU restarts the fault-output (EOUT) timer.

FCCU stops and reinitializes the fault-output (EOUT) timer when all of the following are true:

- If the EOUT signals are in Bi-Stable fault-output mode (CFG[FOM]), and T_{min} (see DELTA_T[DELTA_T]) has expired.
- All faults that caused FCCU to enter or remain in the FAULT state since FCCU started the fault-output (EOUT) timer have been cleared, causing FCCU to return to the NORMAL state.

Prepare the EOUT signals to indicate FCCU's condition

- If the EOUT signals are in Bi-Stable fault-output mode (CFG[FOM]), ensure that the EOUT signals are controlled by FCCU's FSM (CFG[FCCU_SET_CLEAR]).
- Ensure that the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]).

NOTE

If the EOUT signals are in Bi-Stable fault-output mode, you must deactivate and then reactivate the EOUT signals (CFG[FCCU_SET_AFTER_RESET]) to correctly initialize them so they have opposite states.

More about the EOUT interface

Different fault-output modes (protocols) for the fault-output (EOUT) interface are supported (CFG[FOM]):

- Bi-Stable

NOTE

See the chip-specific FCCU information for the fault-output modes supported by this chip.

You can further configure the fault-output modes using the following attributes:

Attribute	Field	Setting used in the example diagrams and tables that follow
Configuration mode	CFG[CM]	Different
Polarity selection	CFG[PS]	For the faulty indication, EOUT1 is high, and EOUT0 is low.

EOUT frequency: This frequency is generated by dividing the CLKSAFE frequency by a fixed factor of 2^{18}.

$$EOUT_{freq} = \frac{CLKSAFE_{freq}}{2^{18}}$$

For example, with a CLKSAFE frequency of 16 MHz, this drives a signal of 61 Hz on EOUT.

In case of a failure event or on software request for EOUT indication, the signal(s) are set to the faulty state for a minimum time (T_{min}), even if software tries to release it before. If software configures the error pins to OK(1), and if a fault comes trying to drive the pin to NOK(0), then priority is given to the fault indication and the error signals indicate NOK, such as an incoming fault is not masked even when software has set the error signal to high. Also, if the error signals are forced to low by software by writing to CFG[FCCU_SET_CLEAR], then the signals shall remain low (or high) for the entire duration of T_{min}. During the T_{min} by a non-software fault, the FCCU FSM moves independently of this signal state (low), and as soon as the timer expires, the pin behavior is dictated by the state in which the FSM finds itself in, and it is not possible to set the signals to OK by software moving FCCU to the CONFIG state, as long as this timer is running. No software intervention is needed to bring the signal from the low state.

Software can bring the pin back to OK state by clearing the faults and waiting for the T_{min} interval to expire, after which the FCCU automatically enters the NORMAL state and the error signal indicates OK.

In case another failure event happens within T_{min} after a first one, the T_{min} counter is restarted.

49.6.5.1 Bi-Stable protocol

The encoding scheme is provided in Table 233 and the related timing diagram is shown in Figure 189.

Table 233. Bi-Stable encoding

Condition	EOUT[1:0] (CFG[PS] is 0)	EOUT[1:0] (CFG[PS] is 1)
Nonfaulty	Static 01	Static 10
Faulty	Static 10	Static 01
Reset	High-impedance (no toggling)[1]	High-impedance (no toggling)
Configuration (CFG[CM] is 1 and CFG[FCCU_SET_AFTER_RESET] is 0)	High-impedance (no toggling)[2]	High-impedance (no toggling)
Configuration (When CFG[CM] is 1 and CFG[FCCU_SET_AFTER_RESET] is 1)	Static 01	Static 10

1. Final value depends on device specific settings at pad level.
2. Ensure that the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]); otherwise, the EOUT signals stay in a high-impedance state after reset lifts.

NOTE

Figure 189 is formatted to display the behavior in all four conditions (reset, nonfaulty, faulty, and configuration), not to imply transitions between one condition and another. In particular, a transition from the faulty condition to the configuration condition is not possible.

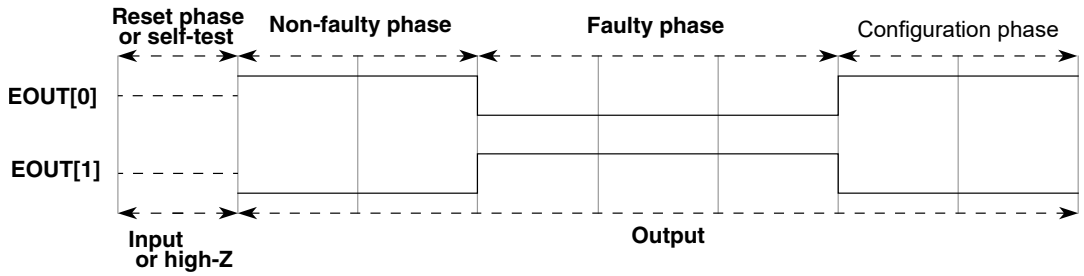


Figure 189. Bi-Stable protocol

49.7 Prepare FCCU for configuration

49.7.1 Introduction

You prepare FCCU for configuration by first configuring the CONFIG state and then putting FCCU in that state.

49.7.2 About preparing FCCU for configuration

When preparing FCCU for configuration, keep the following in mind:

- To put FCCU in CONFIG state, FCCU must be in NORMAL state.
- After FCCU is reset, the configuration is temporarily locked. You must temporarily unlock the configuration before you can put FCCU in the CONFIG state.
- When FCCU is in the CONFIG state, FCCU does not actually save the changes you make to the configuration. To save changes to the configuration, you must manually put FCCU in the NORMAL state. If FCCU automatically leaves the CONFIG state and enters the NORMAL state because the Configuration-state timeout interval (CFG_TO[TO]) expires (called a Configuration-state timeout), FCCU changes the value of the Configuration (CFG) register to its Configuration-state-timeout value and the value of each of the other configuration registers to its reset value. FCCU also changes the value of the Configuration-State Timeout Interval (CFG_TO) register to its reset value. For information on the Configuration-state timeout value, see CFG register bit value at different events. For a list of configuration registers, see Configuration registers.

49.7.3 Configure the CONFIG state

1. Set the Configuration-state timeout interval (CFG_TO[TO]).
2. Enable the Configuration-state-timeout interrupt signal (IRQ_EN[CFG_TO_IEN]), if you want FCCU to request an interrupt when a Configuration-state timeout occurs.

49.7.4 Put FCCU in the CONFIG state

1. Unlock the configuration temporarily (TRANS_LOCK[TRANSKEY]) in Supervisor mode.
2. Check the FCCU status (STAT[STATUS]).
   - If the FCCU status is CONFIG, FCCU is in the CONFIG state. Stop.
   - If the FCCU status is NORMAL, go to step 3.

- If the FCCU status is ALARM or FAULT, go to step 5.

3. Run the OP1 operation (see Run an operation).
4. Check the operation status (CTRL[OPS]).
   - If the operation status is Successful, FCCU is in the CONFIG state. Stop.
   - If the operation status is Aborted, the configuration is probably permanently locked. Go to step 7.
5. Recover all faults (see Fault recovery).
6. Go to step 2.
7. Reset FCCU.

49.8 Configure FCCU

49.8.1 Introduction

You configure FCCU so it functions according to the needs of your particular application.

49.8.2 About configuring FCCU

When configuring FCCU:

- If you enable a non-critical fault channel but disable all reactions for that channel, FCCU changes state when necessary but does not perform any reaction because reactions are disabled. If you enable reactions for a non-critical fault channel but disable that channel, and FCCU is in the NORMAL state when a fault occurs on the channel, FCCU does not enter the ALARM or FAULT state and therefore does not perform any reaction.

49.8.3 Configure the non-critical fault channels

For each non-critical fault channel that you want FCCU to monitor:

1. Set the recovery type (NCF_CFGa[NCFCn]).
2. Enable at least one type of Fault-state reaction:
   - Chip functional reset (NCFS_CFGa[NCFSCn])
   - Non-maskable interrupt (NMI_ENa[NMIENn])
   - EOUT signaling (EOUT_SIG_ENa[EOUTENn])

   NOTE

   If you enable the chip functional reset as the type of Fault-state reaction for a channel, enable at least one other type of Fault-state reaction for the channel or enable the ALARM state (step 4) for the channel.

3. Set the Alarm-state timeout interval (NCF_TO[TO]), if you plan to enable the ALARM state for any non-critical fault channel.

   NOTE

   Ensure that the Alarm-state timeout interval is less than the FOSU module's timeout interval; otherwise, FOSU generates a chip reset every time a fault occurs on the channel. The FOSU timeout interval (FOSU_COUNT) is a chip-specific value. See the chip-specific FCCU information.

4. Enable the ALARM state (NCF_TOEa[NCFTOEn]) for any non-critical fault channel for which you want FCCU to enter the ALARM state before entering the FAULT state.
5. Enable the Alarm-state reaction (IRQ_ALARM_ENa[IRQENn]) for each non-critical fault channel for which you enabled the ALARM state.
6. Enable the corresponding NCF_Ea[NCFEn] field for each non-critical fault channel that you want FCCU to monitor.

49.9 Put FCCU in the NORMAL state

49.9.1 Introduction

You put FCCU in the NORMAL state to save changes to the configuration, and to allow FCCU to enter the ALARM or FAULT state when a fault occurs on an enabled fault channel.

49.9.2 About putting FCCU in NORMAL state

When putting FCCU in the NORMAL state:

- If you attempt to lock the configuration while FCCU is in CONFIG state, FCCU does not actually lock the configuration until FCCU leaves CONFIG state—that is, either you put FCCU in the NORMAL state, or FCCU puts itself in the NORMAL state because the Configuration-state timeout interval (CFG_TO[TO]) expires.
- After you permanently lock the configuration, you must reset FCCU before you can put FCCU in the CONFIG state.

49.9.3 Put FCCU in the NORMAL state

1. Check the FCCU status (STAT[STATUS]).
   - If the FCCU status is NORMAL, go to step 6.
   - If the FCCU status is CONFIG, go to step 2.
   - If the FCCU status is ALARM or FAULT, go to step 4.
2. Run the OP2 operation (see Run an operation).
3. Check the operation status (CTRL[OPS]).
   - If the operation status is Successful, FCCU is in the NORMAL state. Go to step 6.
   - If the operation status is Aborted, go to step 2.
4. Recover all faults (see Fault recovery).
5. Go to step 1.
6. Lock the configuration if you want to prevent any changes to it:
   - To require a key to unlock the configuration, from Supervisor mode, temporarily lock the configuration (TRANS_LOCK[TRANSKEY]).
   - To require a reset of FCCU to unlock the configuration, from Supervisor mode, permanently lock the configuration (PERMNT_LOCK[PERMNTKEY]).

     The configuration is permanently locked until FCCU is reset.

49.10 Manage faults

49.10.1 Introduction

After saving changes to the configuration, you are ready to use FCCU to manage faults.

49.10.2 Determine if there are any unrecovered non-critical faults

Check the unrecovered-fault indicators for the non-critical faults (NCF_Sa[NCFSn]).

49.10.3 Recover a software-recoverable non-critical fault

1. Resolve the source of the software-recoverable non-critical fault.
2. Unlock the NCF_Sa registers (NCFK[NCFK]) using a 32-bit write.

3. Initiate clearing of the unrecovered-fault indicator for the software-recoverable non-critical fault (NCF_Sa[NCFSn]) using a 32-bit write.

   FCCU initiates the OP12 operation.

   NOTE

   If you want to clear multiple unrecovered-fault indicators and those indicators reside in different NCF_Sa registers, you must perform steps 2 and 3 for each individual register.

4. Check the operation status (CTRL[OPS]).
   - If the operation status is In Progress, go to step 4.
   - If the operation status is Successful, go to step 5.
   - If the operation status is Aborted, go to step 2.
5. Check the unrecovered-fault indicator for the software-recoverable non-critical fault (NCF_Sa[NCFSn]).
   - If the indicator indicates no unrecovered fault, the fault has been recovered. Stop.
   - If the indicator still indicates an unrecovered fault, go to step 2.

49.10.4 Clear the freeze-status indicators

1. Run the OP13 operation (see Run an operation).
2. Check the operation status (CTRL[OPS]).
   - If the operation status is Successful, stop .
   - If the operation status is Aborted, go to step 1.

49.11 Run operations

49.11.1 Introduction

You run operations to perform actions such as putting FCCU in the CONFIG state or setting the operation status to Idle. For a complete list of operations you can run, see CTRL[OPR].

49.11.2 About running operations

When running operations:

- FCCU ignores any operations initiated while the operation status is In Progress.
- Certain operations must be unlocked before you can initiate them. After you initiate them, they are locked again.

49.11.3 Run an operation

1. Check the operation status (CTRL[OPS]).
2. Go to step 1 if the operation status is In Progress.
3. Unlock the operation (CTRLK[CTRLK]) using a 32-bit write, if the operation must be unlocked before you can initiate it.

   NOTE

   The Control Key (CTRLK) register used in this step and the Control (CTRL) register used in the next step must be written with consecutive instructions. Do not use read-modify-write instructions, such as bit-field instructions, to modify these registers.

4. Initiate the operation (CTRL[OPR]) using a 32-bit write.
5. Check the operation status (CTRL[OPS]).

6. Go to step 5 if the operation status is In Progress.

The operation status is now Idle (OP15 only), Aborted, or Successful.

49.12 FOSU

The FOSU provides a supervision of the primary fault notification path by analyzing FCCU's behavior for correctness. It waits for any reaction of the FCCU in a fixed time window after a fault is signaled.

The intention of the FOSU is to provide a secondary fault reaction path in most cases when the FCCU fails but not to needlessly propagate a fault which is already handled by the FCCU in a full chip reset. Only a failed primary fault reaction (that is, FCCU's failure) is a reason for the secondary reaction to take over (and generate a destructive reset request).

There is a 'do nothing' input coming from FCCU which indicates that the FCCU is programmed for no reaction for ALL FAULTS. It is a "static" input in the sense that it does not change after FCCU configuration. The FOSU masks the incoming faults with the 'do nothing' control from the FCCU, meaning that a fault is not captured by the FOSU if the 'do nothing' signal is asserted (that is, a disabled fault). There is no minimum pulse width requirement on the fault indication other than what is required by the technology, which is the same as that of the FCCU. FOSU does not monitor FCCU for the case of faults occurring during the CONFIG state.

The FOSU contains a timer with a duration of FOSU_COUNT, driven by CLKSAFE. The timer is initialized and started on any captured, enabled fault. While the timer is running, any subsequent captured fault will neither restart nor reinitialize the timer. The timer is stopped when the FCCU shows any of the following reactions (the FOSU does not check whether the reaction is the configured one for the faults which occurred):

- Reset: short functional reset
- IRQ (triggered by ALARM state)
- NMI
- Error out triggered (by FCCU or by SW)

When the timer is stopped, the fault capture logic is cleared to ensure that the timer is not restarted because of faults still 'stuck' in the capture logic. The timer is then restarted by the next new failure indication. When the timer expires, the FOSU's failure indicator output is asserted after it ensures that the fault is enabled and the static "fccu program to do nothing" signal is deasserted. This is because FCCU uses settings after it exits CONFIG state, even if fault captured before the exit.

The FOSU's failure indicator output is connected to one of the MC_RGM's 'destructive' reset inputs, so its assertion will cause a reset sequence to be initiated starting at DEST0. The FOSU module is reset with the same reset as is used by the FCCU. When this reset is asserted, the FOSU's capture logic is cleared, its timer is kept stopped and in a non-expired state, and its failure indicator output is deasserted.

NOTE

FOSU is triggered on assertion of enabled fault. In case the triggering fault is disabled, FOSU times out without reaction.

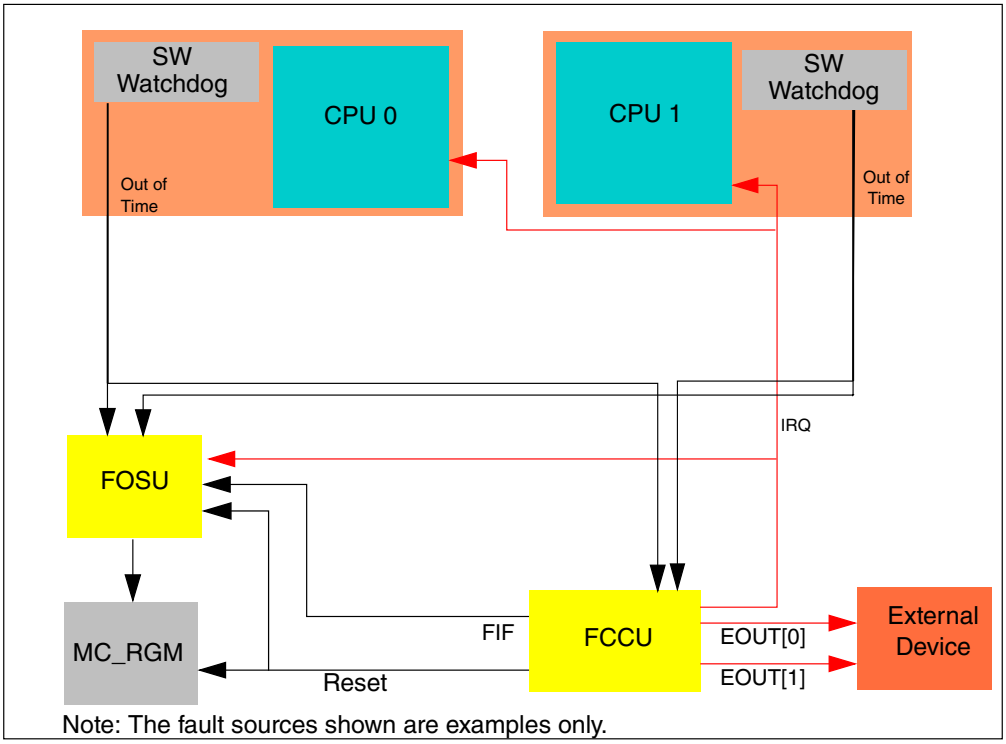


Figure 190. FOSU connections to the FCCU and MC_RGM

49.13 Use cases and limitations

Configuration guidelines

Follow these guidelines to configure FCCU:

- If you want FCCU to react to a fault on a non-critical fault channel:
  - Enable the channel (Non-critical Fault Enable (NCF_E0)).
  - Enable at least one type of Fault-state reaction for the channel: chip reset (Non-critical Fault-State Configuration (NCFS_CFG0)), fault-output (EOUT) signaling (Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0)), or non-maskable interrupt (Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0)).
  - If you enable chip reset as the type of Fault-state reaction for the channel (Non-critical Fault-State Configuration (NCFS_CFG0)), enable either ALARM state (Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0)) or at least one other type of Fault-state reaction for the channel: fault-output (EOUT) signaling (Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0)) or non-maskable interrupt (Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0)).
  - If you enable ALARM state for the channel (Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0)), enable the Alarm-state reaction (Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0)).
  - If you enable ALARM state for the channel (Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0)), make sure the Alarm-state timer interval (Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)) is less than the FOSU module's timer interval; otherwise, FOSU generates a chip reset every time a fault occurs on the channel. The FOSU timer interval (FOSU_COUNT) is chip-specific. See the chip-specific FCCU information.

NOTE

If you enable a non-critical fault channel but disable all reactions for that channel, FCCU changes state if necessary but doesn't perform any reaction because reactions are disabled. If you enable reactions for a non-critical fault channel but disable that channel, and FCCU is in NORMAL state when a fault occurs on the channel, FCCU doesn't enter ALARM or FAULT state and therefore doesn't perform any reaction.

Recommendations to configure FCCU

1. After a power on, or 'destructive' reset (when initiated by the assertion of the chip reset pin, RESET_B), where both system and FCCU are reset, the following steps could be followed to configure FCCU:
    a. Check and clear any pending fault status
    b. Verify FCCU is in NORMAL state, else repeat step(a) above
    c. Configure FCCU
2. After any 'functional' reset of the system, arising out of a reset request from FCCU or other sources, the following steps could be followed to reconfigure FCCU:
    a. If active, wait for the Error out T_{min} to expire
    b. Check and clear fault status
    c. Error pin moves to "non faulty" state, once fault status is cleared and T_{min} expires
    d. Verify FCCU is in NORMAL state, else repeat step(a) above
    e. Read and verify value in NCF_En
    f. Reconfigure FCCU, if necessary

49.14 Register descriptions

49.14.1 FCCU register descriptions

The FCCU registers are listed in the table below. Any address offset not explicitly mentioned in this table is reserved.

The FCCU supports word (32-bit), half-word (16-bit), and byte (8-bit) read and write accesses.

Follow these register-access guidelines:

- Do not read from or write to any addresses that are not shown in the following table. Doing so may or may not result in a transfer error.
- Do not write to any of the configuration registers unless FCCU is in the CONFIG state. Doing so results in a transfer error.
- Do not write to the Transient Configuration Lock (TRANS_LOCK) or Permanent Configuration Lock (PERMNT_LOCK) registers unless your code runs in the Supervisor mode. Doing so results in a transfer error.

For each possible NCF failure source, a different reaction—including no reaction—is configurable through the use of NMI, IRQ, and short reset selection registers. It is not possible for a single event upset to switch off all reactions on failures as implementation is per fault source (but it will be possible to switch them all off by SW if intended). Failures themselves are not able to disable all reactions and indications.

The FCCU is not reset by short functional resets.

49.14.1.1 FCCU memory map

FCCU base address: 4038_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CTRL)	32	RW	0000_00C0h
4h	Control Key (CTRLK)	32	W	0000_0000h
8h	Configuration (CFG)	32	RW	0000_0000h
1Ch	Non-critical Fault Configuration (NCF_CFG0)	32	RW	0000_00FFh
4Ch	Non-critical Fault-State Configuration (NCFS_CFG0)	32	RW	0000_0000h
80h	Non-critical Fault Status (NCF_S0)	32	RW	0000_0000h
90h	Non-critical Fault Key (NCFK)	32	W	0000_0000h
94h	Non-critical Fault Enable (NCF_E0)	32	RW	0000_0000h
A4h	Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0)	32	RW	0000_00FFh
B4h	Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)	32	RW	0003_A980h
B8h	Configuration-State Timeout Interval (CFG_TO)	32	RW	0000_0005h
BCh	IO Control (EINOUT)	32	RW	See section
C0h	Status (STAT)	32	R	0000_0010h
C4h	Normal-to-Alarm Freeze Status (N2AF_STATUS)	32	R	0000_0000h
C8h	Alarm-to-Fault Freeze Status (A2FF_STATUS)	32	R	0000_0000h
CCh	Normal-to-Fault Freeze Status (N2FF_STATUS)	32	R	0000_0000h
D0h	Fault-to-Alarm Freeze Status (F2AF_STATUS)	32	R	0000_0000h
DCh	Non-critical Fault Fake (NCFF)	32	RW	0000_0000h
E0h	IRQ Status (IRQ_STAT)	32	RW	0000_0000h
E4h	IRQ Enable (IRQ_EN)	32	RW	0000_0000h
F0h	Transient Configuration Lock (TRANS_LOCK)	32	RW	0000_0000h
F4h	Permanent Configuration Lock (PERMNT_LOCK)	32	RW	0000_0000h
F8h	Delta T (DELTA_T)	32	RW	0000_0000h
FCh	Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0)	32	RW	0000_0000h
10Ch	Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0)	32	RW	0000_0000h
11Ch	Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0)	32	RW	0000_0000h
12Ch	Alarm-State Timer (TMR_ALARM)	32	R	0003_A980h
134h	Configuration-State Timer (TMR_CFG)	32	R	000F_FFFFh
138h	Fault-Output Timer (TMR_ETMR)	32	R	0000_0000h

49.14.1.2 Control (CTRL)

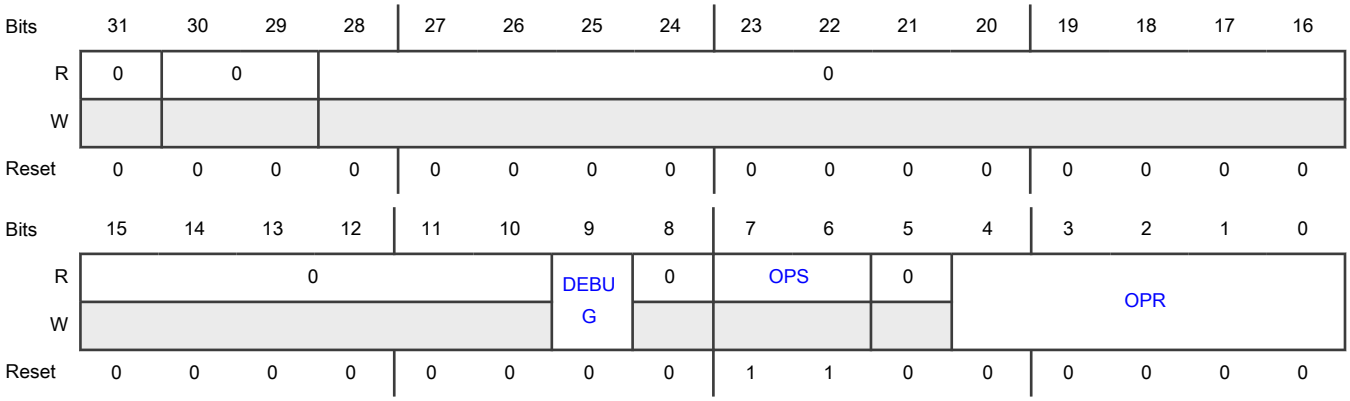
Offset

Register	Offset
CTRL	0h

Function

Initiates and indicates the status of operations—and enables the Debug mode.

Diagram



Fields

Field	Function
31 —	Reserved
30-29 —	Reserved
28-10 —	Reserved
9 DEBUG	Debug Mode Enable Specifies whether the Debug mode is enabled. If so, FCCU enters the Debug mode when the Debug signal is asserted. When FCCU enters the Debug mode, it halts operation and remains in the state it was in before it entered this mode. **NOTE** FOSU does not halt when FCCU enters the Debug mode. Therefore, FOSU can still cause a reset if a fault occurs while FCCU is in the Debug mode. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
8 —	Reserved
7-6 OPS	Operation Status This field can be read and cleared (via OP15 operation) by the software. 00b - Idle 01b - In progress 10b - Aborted 11b - Successful
5 —	Reserved
4-0 OPR	Operation Run Initiates operations that perform actions such as putting FCCU in the CONFIG state or setting the operation status to Idle. For information on how to run operations, see Run operations. FCCU ignores any write to this field while the operation status (CTRL[OPS]) is "In progress". After completion of an operation, FCCU sets this field to OP0. The following events result in an operation status (CTRL[OPS]) of "Aborted": • Writing to a NCF_Sa register (which automatically initiates the OP12 operation) without first successfully unlocking the register (NCFK[NCFK]) • Initiating an OP1 operation when FCCU is not in the NORMAL state or the configuration is locked • Initiating an OP1, or OP2, operation without first unlocking the operation (CTRLK[CTRLK]) 00000 OP0—No operation 00001 OP1—Applies only when the configuration is unlocked, when FCCU is in the NORMAL state, and immediately after you unlock the operation (CTRLK[CTRLK]). Put FCCU in the CONFIG state. 00010 OP2—Applies only immediately after you unlock the operation (CTRLK[CTRLK]). Put FCCU in the NORMAL state. 00011 Reserved 00100 Reserved 00101 Reserved 00110 Reserved 00111 Reserved 01000 Reserved 01001 Reserved 01010 Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01011 Reserved 01100 OP12—Do not initiate this operation; it is automatically initiated by the FCCU. A NCF_Sa register status clear operation is in progress. 01101 OP13—Clear the freeze status registers. 01110 OP14—Do not initiate this operation; it is automatically initiated by the FCCU. A Configuration-state timeout is in progress. For more information, see Configuration registers. 01111 OP15—Set the operation status (CTRL[OPS]) to Idle. 10000 Reserved 10001 Reserved 10010 Reserved 10011 Reserved 10100 Reserved 10101—11110 Forbidden. Writing any of these values returns an operation status (CTRL[OPS]) of "Aborted" with no side effect. 11111 Reserved

49.14.1.3 Control Key (CTRLK)

Offset

Register	Offset
CTRLK	4h

Function

See CTRLK[CTRLK].

Diagram

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

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

Fields

Field	Function
31-0 CTRLK	Locked-Operation Control Key Writable only with a 32-bit write. Unlocks locked operations (CTRL[OPR]) so you can initiate them. For information on how to unlock locked operations before you initiate them, see Run an operation. **NOTE** • You must initiate an operation in the FCCU register access that immediately follows the one that unlocks it; otherwise, the operation is again locked. • Reading from this register always returns the value 0000_0000h. Operations not listed here are not locked and do not need to be unlocked. 9137_56AFh: Unlock OP1. 825A_132Bh: Unlock OP2. Any other value: Do nothing.

49.14.1.4 Configuration (CFG)

Offset

Register	Offset
CFG	8h

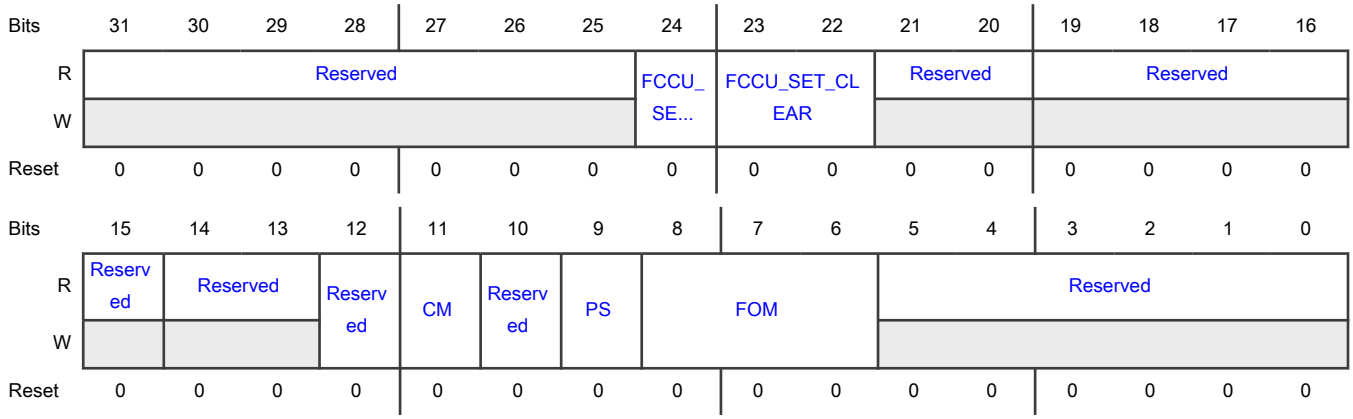
Function

Writable only when FCCU is in the CONFIG state. Changed by FCCU to another value when the chip resets FCCU, a Configuration-state timeout occurs, or you run an OP31 operation. See CFG register bit value at different events for more information. Specifies the global configuration for FCCU.

NOTE

If you specify a new value for any of the fields in this register that affect the EOUT signals while the fault-output (EOUT) timer is running (FCCU is indicating a fault on the EOUT signals), FCCU does not use the new settings you specified until after the fault-output (EOUT) timer expires (FCCU stops indicating a fault on the EOUT signals).

Diagram



Fields

Field	Function
31-25 —	Reserved
24 FCCU_SET_AFTER_RESET	Fault-Output (EOUT) Activate For fault-output (EOUT) signaling, controls whether the EOUT signals are active. 0b - Inactive (the EOUT signals are in a high-impedance state) 1b - Active (the EOUT signals indicate FCCU's condition)
23-22 FCCU_SET_CLEAR	Fault-Output (EOUT) Control Applies only to Bi-Stable fault-output mode (CFG[FOM]) and when the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]). Controls whether the fault-output (EOUT) signals are managed by FCCU's FSM. 00b - Controlled by the FSM 01b - Always low 10b - Controlled by the FSM 11b - High until a fault occurs on a channel, regardless of whether that fault is disabled; thereafter, controlled by the FSM. Note: FCCU ignores an attempt to write this value if the fault-output (EOUT) timer is already running.
21-20 —	Reserved
19-16 —	Reserved
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-13 —	Reserved
12 —	Reserved Always write the reset value to this field.
11 CM	Fault-Output (EOUT) Configuration-Indication Mode For fault-output (EOUT) signaling, this field controls whether the configuration indication is the same as the nonfaulty indication. 0b - Different 1b - Same
10 —	Reserved
9 PS	Fault-Output (EOUT) Polarity Selection Applies to fault-output (EOUT) signaling and controls the polarity of the signals for fault-output mode indications that hold the signals low or high (versus toggling them or placing them in a high-impedance state). Applies only to Bi-Stable fault-output mode (for all indications). 0b - For the faulty indication, EOUT1 is high, and EOUT0 is low. 1b - For the faulty indication, EOUT1 is low, and EOUT0 is high.
8-6 FOM	Fault-Output (EOUT) Mode For fault-output (EOUT) signaling, controls the protocol of the signaling. 000b - Reserved 001b - Reserved 010b - Bi-Stable 011b - Reserved 100b - Reserved 101b - Test 0 (controlled by the EINOUT register; EOUT1 is an output; EOUT0 is an input) 110b - Test 1 (controlled by the EINOUT register; EOUT1 and EOUT0 are both outputs) 111b - Test 2 (controlled by the EINOUT register; EOUT1 is an input; EOUT0 is an output)
5-0 —	Reserved

49.14.1.5 Non-critical Fault Configuration (NCF_CFG0)

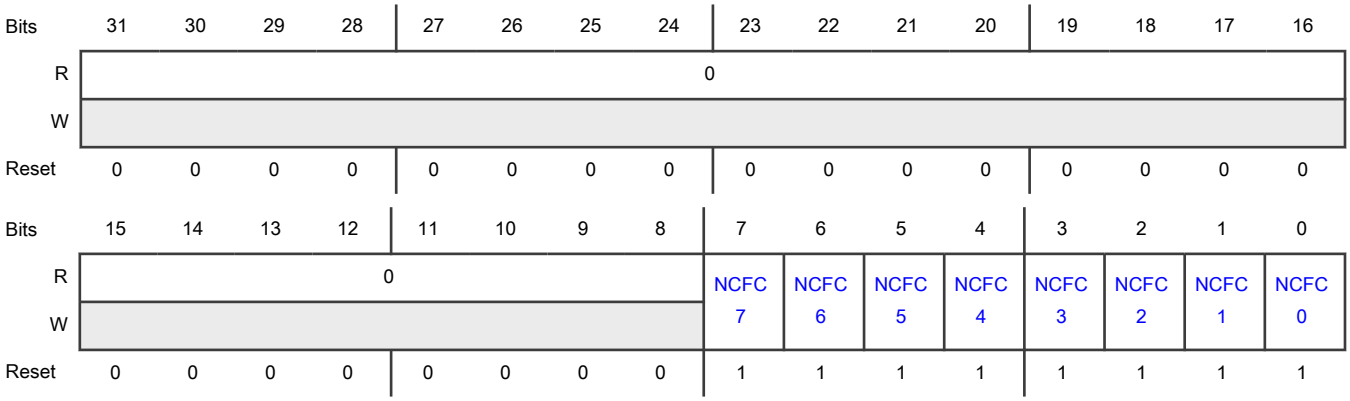
Offset

Register	Offset
NCF_CFG0	1Ch

Function

See NCF_CFGa[NCFCn].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 NCFCn	Non-critical Fault Configuration n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls the recovery type (HW or SW) of the associated non-critical fault channel (n). For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. **NOTE** Configure a non-critical fault channel as hardware-recoverable only if the source continues to indicate a fault on the fault channel's input (NCFn) until the condition that caused the fault is no longer true; otherwise, configure the non-critical fault channel as software-recoverable. 0b - Hardware-recoverable 1b - Software-recoverable

49.14.1.6 Non-critical Fault-State Configuration (NCFS_CFG0)

Offset

Register	Offset
NCFS_CFG0	4Ch

Function

See NCFS_CFGa[NCFSCn].

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NCFSC7		NCFSC6		NCFSC5		NCFSC4		NCFSC3		NCFSC2		NCFSC1		NCFSC0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-14: NCFSC7 13-12: NCFSC6 11-10: NCFSC5 9-8: NCFSC4 7-6: NCFSC3 5-4: NCFSC2 3-2: NCFSC1 1-0: NCFSC0	Non-critical Fault-State Configuration n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the chip functional reset is enabled as a Fault-state reaction for the associated non-critical fault channel (n). When the chip functional reset is enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to assert the rst_sfunc_b signal when FCCU enters the FAULT state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. 00b - Disabled 01b - Enabled (rst_sfunc_b) (short) 10b - Reserved 11b - Disabled

49.14.1.7 Non-critical Fault Status (NCF_S0)

Offset

Register	Offset
NCF_S0	80h

Function

See NCF_Sa[NCFSn].

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								NCFS 7	NCFS 6	NCFS 5	NCFS 4	NCFS 3	NCFS 2	NCFS 1	NCFS 0
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 NCFSn	Non-critical Fault Status n Indicates whether there is an unrecovered fault on the associated non-critical fault channel (n). **NOTE** To recover a software-recoverable non-critical fault, which includes clearing its unrecovered-fault indicator, see Recover a software-recoverable non-critical fault. FCCU clears the unrecovered-fault indicator for a hardware-recoverable non-critical fault automatically when the source no longer indicates a fault on the fault channel's input signal; if you attempt to clear the unrecovered-fault indicator for a hardware-recoverable non-critical fault, FCCU does not clear the indicator and does not indicate an error. 0b - No unrecovered fault 1b - Unrecovered fault

49.14.1.8 Non-critical Fault Key (NCFK)

Offset

Register	Offset
NCFK	90h

Function

See NCFK[NCFK].

Diagram

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

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

Fields

Field	Function
31-0 NCFK	Non-critical Fault Key Writable only with a 32-bit write. Unlocks the NCF_Sa registers so you can write to them while recovering a software-recoverable non-critical fault. For information on how to unlock the NCF_Sa registers before writing to them, see Recover a software-recoverable non-critical fault. **NOTE** • You must write to one of the NCF_Sa registers immediately after unlocking it (that is, in the FCCU register access that immediately follows the one that unlocks them); otherwise the registers are again locked. If you want to write to multiple NCF_Sa registers, you must unlock each register immediately before you write to it. • Reading from this register always returns the value 0000_0000h. AB34_98FEh: Unlock. Any other value: Do nothing.

49.14.1.9 Non-critical Fault Enable (NCF_E0)

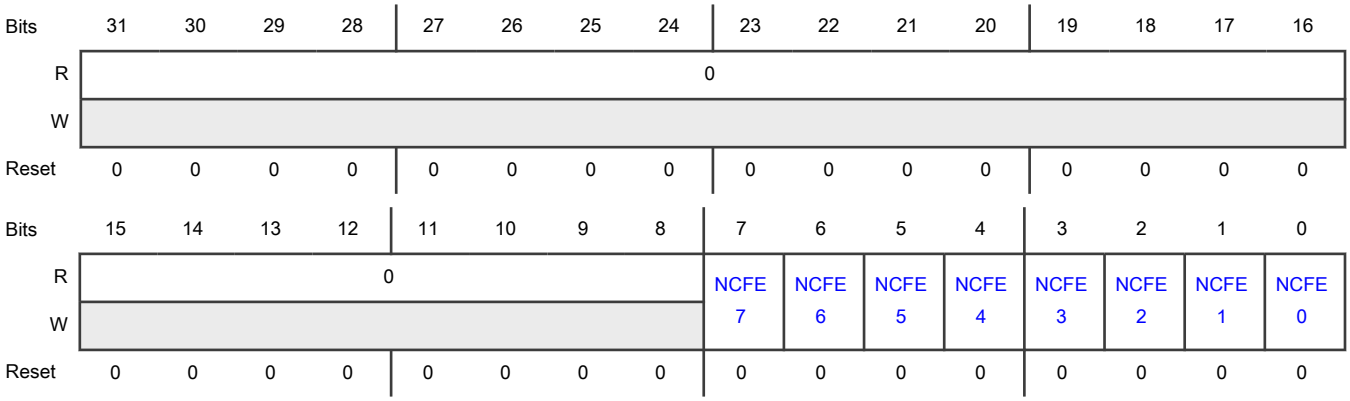
Offset

Register	Offset
NCF_E0	94h

Function

See NCF_Ea[NCFEn].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 NCFEn	Non-critical Fault Enable n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the associated non-critical fault channel (n) is enabled. When a non-critical fault channel is enabled, a fault on that channel causes FCCU to leave the NORMAL state and enter the FAULT state (or ALARM state if enabled for the channel). For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. 0b - Disabled 1b - Enabled

49.14.1.10 Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0)

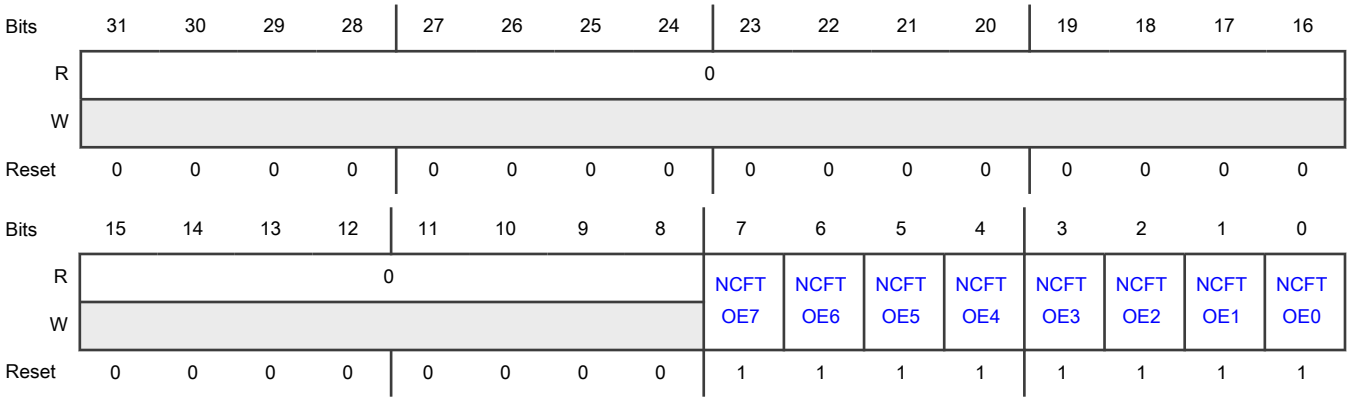
Offset

Register	Offset
NCF_TOE0	A4h

Function

See NCF_TOEa[NCFTOEn].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 NCFTOEn	Non-critical-Fault Alarm-State Timeout Enable n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the ALARM state is enabled for the associated non-critical fault channel (n). When the ALARM state is enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to leave the NORMAL state and enter the ALARM state instead of FAULT state. If the fault is not recovered within the Alarm-state timeout interval, then FCCU leaves the ALARM state and enters the FAULT state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. 0b - Disabled 1b - Enabled

49.14.1.11 Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)

Offset

Register	Offset
NCF_TO	B4h

Function

See NCF_TO[TO]

Diagram

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

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

Fields

Field	Function
31-0 TO	Non-critical-Fault Alarm-State Timeout Interval Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls the maximum amount of time that FCCU can be in the ALARM state ($T_{Max\ Alarm}$) according to this equation: $T_{Max\ Alarm} = TO \times T_{CLKSAFE}$ where $T_{CLKSAFE}$ is the safe-clock period. If FCCU enters the ALARM state (because a fault occurs on an enabled non-critical fault channel for which the ALARM state is enabled) and this timeout interval expires (called an Alarm-state timeout), then FCCU leaves the ALARM state and enters the FAULT state. **NOTE** Make sure the Alarm-state timeout interval is less than the FOSU module's timeout interval; otherwise, a fault that occurs while FCCU is in the ALARM state can cause FOSU to generate a chip reset. The FOSU timeout interval (FOSU_COUNT) is chip-specific. See the chip-specific FCCU information.

49.14.1.12 Configuration-State Timeout Interval (CFG_TO)

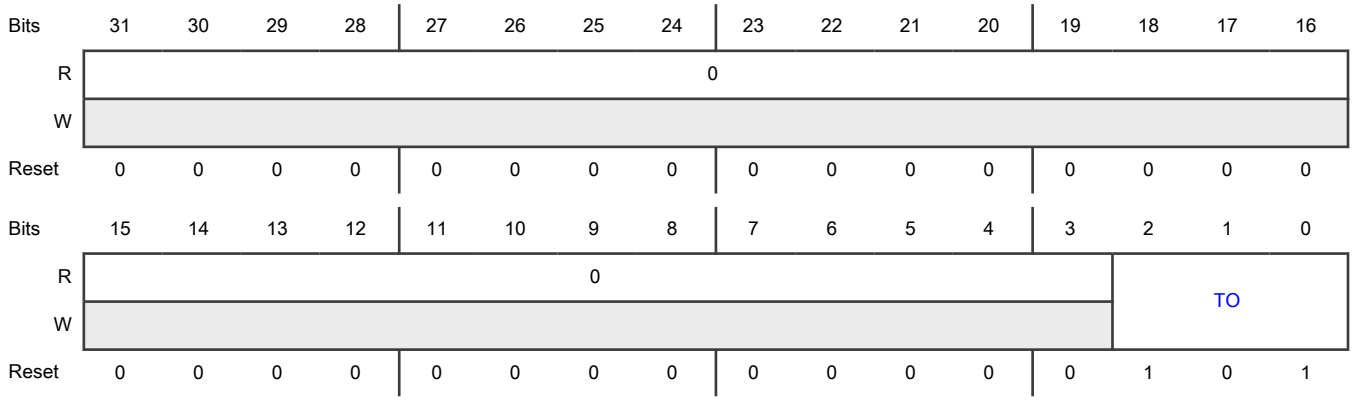
Offset

Register	Offset
CFG_TO	B8h

Function

See CFG_TO[TO]

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 TO	Configuration-State Timeout Interval Writable only when FCCU is in the NORMAL, ALARM, or FAULT state (not in the CONFIG state). Changed by FCCU to its reset value when a Configuration-state timeout occurs. Not accessible while a Configuration-state timeout (OP14 operation) is in progress. Controls the maximum amount of time that FCCU can be in the CONFIG state ($T_{\text{Max configuration}}$) according to this equation: $T_{\text{Max configuration}} = T_{\text{CLKSAFE}} \times 2^{(\text{TO} + 13)}$ where T_{CLKSAFE} is the safe-clock period. If you put FCCU in the CONFIG state and this timeout interval expires (called a Configuration-state timeout), then FCCU: • Automatically leaves the CONFIG state and enters the NORMAL state • Changes the value of the Configuration (CFG) register to its Configuration-state-timeout value and the value of each of the other configuration registers to its reset value. For information on the Configuration (CFG) register's Configuration-state-timeout value, see CFG register bit value at different events. For a list of configuration registers, see Configuration registers.

49.14.1.13 IO Control (EINOUT)

Offset

Register	Offset
EINOUT	BCh

Function

The EINOUT register allows the following operations typically in the NORMAL state:

- To control the EOUT[1] output level when the FCCU is configured in "Test1" or "Test0" fault output mode (CFG[FOM])

- To control the EOUT[0] output level when the FCCU is configured in "Test1" or "Test2" fault output mode (CFG[FOM])
- to observe the state of signals at EIN[1:0] pins

The following table shows Bi-Stable encoding.

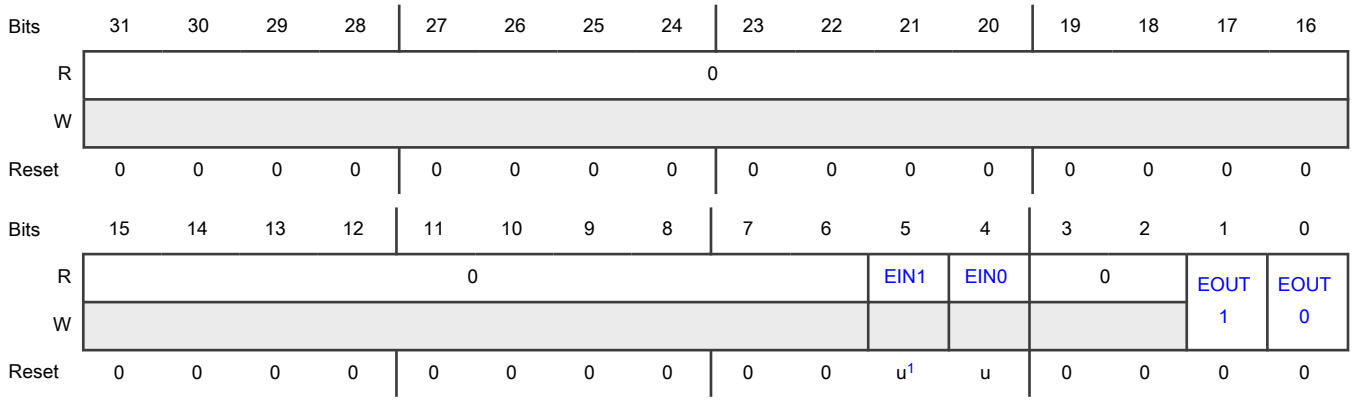
Table 234. Bi-Stable encoding

Mode = CFG[FOM]	EOUT[0]	EOUT[1]
Test1	output	output
Test2	output	input
Test0	input	output

NOTE

Because of the resynchronization stage of the EOUT interface, there is a latency of a few CLKSAFE cycles following a write/read operation of the EINOUT register.

Diagram



1. Reset value varies as per the corresponding EIN signal.

Fields

Field	Function
31-6 —	Reserved
5 EIN1	Error Input 1 Applies only when the EOUT signals are active (FCCU_SET_AFTER_RESET). Indicates the state of the EIN1 signal. 0b - Low 1b - High
4 EIN0	Error Input 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Applies only when the EOUT signals are active (FCCU_SET_AFTER_RESET). Indicates the state of the EIN0 signal. 0b - Low 1b - High
3-2 —	Reserved
1 EOUT1	EOUT1 Error out 1 (significant only if the CFG.FOM = Test1 or Test0 => EOUT[1] configured in output mode). The EOUT1 set/clear the respective EOUT[1] output signal if CFG.FOM = 110 or 101, otherwise it is a "don't-care" value. **NOTE** When the configuration watchdog timer expires, FCCU changes the value of this field to its reset value. 0b - force EOUT[1] = 0 1b - force EOUT[1] = 1
0 EOUT0	EOUT0 Error out 0 (significative only if the CFG.FOM = Test1 or Test2 => EOUT[0] configured in output mode). The EOUT0 set/clear the respective EOUT[0] output signal if CFG.FOM = 110 or 111, otherwise it is a "don't care" value. **NOTE** When the configuration watchdog timer expires, FCCU changes the value of this field to its reset value. 0b - force EOUT[0] = 0 1b - force EOUT[0] = 1

49.14.1.14 Status (STAT)

Offset

Register	Offset
STAT	C0h

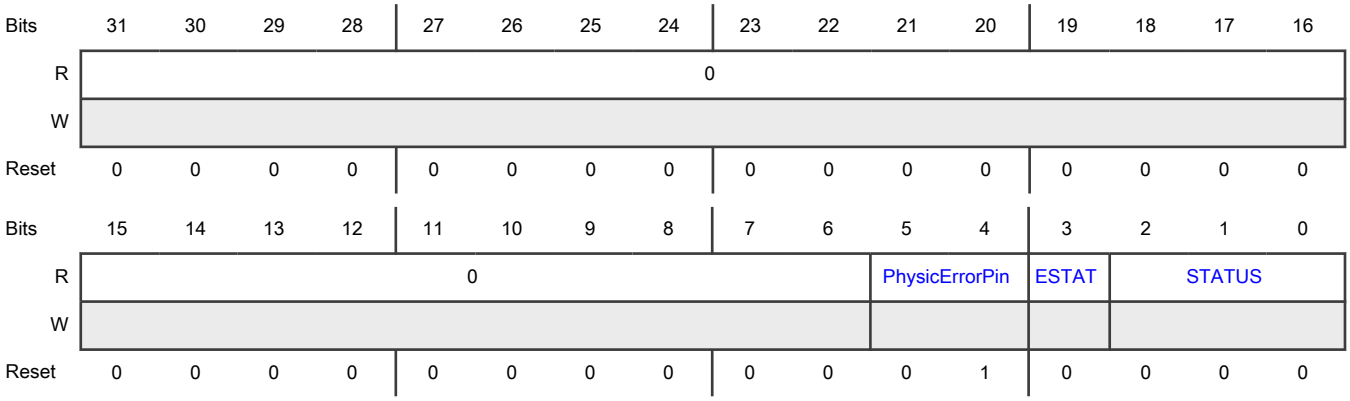
Function

This register indicates the following:

- States that FCCU is driving on the EOUT signals
- Whether FCCU is in a faulty condition

- Current state of FCCU

Diagram



Fields

Field	Function
31-6 —	Reserved
5-4 PhysicErrorPin	EOUT Signal States Applies only when the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]). Indicates the states that FCCU is driving on the EOUT signals. 00b - EOUT1 is low; EOUT0 is low. 01b - EOUT1 is low; EOUT0 is high. 10b - EOUT1 is high; EOUT0 is low. 11b - EOUT1 is high; EOUT0 is high.
3 ESTAT	FCCU Faulty Condition Indicates whether FCCU is in faulty condition (as indicated by the EOUT signals). For more information, see The FCCU conditions. 0b - Not in faulty condition (in nonfaulty or configuration condition) 1b - In faulty condition
2-0 STATUS	FCCU State Indicates the current state of FCCU 000b - NORMAL 001b - CONFIG 010b - ALARM 011b - FAULT 100b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	101b - Reserved 110b - Reserved 111b - Reserved

49.14.1.15 Normal-to-Alarm Freeze Status (N2AF_STATUS)

Offset

Register	Offset
N2AF_STATUS	C4h

Function

See N2AF_STATUS[NAFS].

Diagram

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

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

Fields

Field	Function
31-8 —	Reserved
7-0 NAFS	Normal-to-Alarm Freeze Status Used only for testing and debugging. Indicates whether FCCU left the NORMAL state and entered the ALARM state since the last time this register was cleared and, if so, which non-critical fault caused FCCU to do so.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00h: No Normal-to-Alarm-state transition (cleared) 01h: NCF0 10h: NCF1 ... 7Fh: NCF126 80h: NCF127 ... FFh: Multiple Normal-to-Alarm-state transitions

49.14.1.16 Alarm-to-Fault Freeze Status (A2FF_STATUS)

Offset

Register	Offset
A2FF_STATUS	C8h

Function

Used only for testing and debugging. Indicates whether FCCU left the ALARM state and entered the FAULT state since the last time this register was cleared and, if so, which type of fault caused FCCU to do so.

NOTE

To clear this register and the other freeze status registers, see Clear the freeze-status indicators.

Diagram

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

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

Fields

Field	Function
31-10 —	Reserved
9-8 AF_SRC	Alarm-to-Fault Source Used only for testing and debugging. Indicates the type of fault that caused FCCU to leave the ALARM state and enter the FAULT state since the last time this register was cleared. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00b - No Alarm-to-Fault-state fault 01b - Reserved 10b - Non-critical fault 11b - Multiple Alarm-to-Fault-state faults
7-0 AFFS	Alarm-to-Fault Freeze Status Used only for testing and debugging. Indicates whether FCCU left the ALARM state and entered the FAULT state since the last time this register was cleared and, if so, which fault caused FCCU to do so. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00h: No Alarm-to-Fault-state transition (cleared) 01h: NCF0 (due to an Alarm-state timeout) 10h: NCF1 (due to an Alarm-state timeout) ... 7Fh: NCF126 (due to an Alarm-state timeout) 80h: NCF127 (due to an Alarm-state timeout) ... FFh: Multiple Alarm-to-Fault-state transitions

49.14.1.17 Normal-to-Fault Freeze Status (N2FF_STATUS)

Offset

Register	Offset
N2FF_STATUS	CCh

Function

Used only for testing and debugging. Indicates whether FCCU left the NORMAL state and entered the FAULT state since the last time this register was cleared and, if so, which type of fault caused FCCU to do so.

NOTE

To clear this register and the other freeze status registers, see Clear the freeze-status indicators.

Diagram

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

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

Fields

Field	Function
31-10 —	Reserved
9-8 NF_SRC	Normal-to-Fault Source Used only for testing and debugging. Indicates the type of fault that caused FCCU to leave the NORMAL state and enter the FAULT state since the last time this register was cleared. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00b - No Normal-to-Fault-state fault 01b - Reserved 10b - Non-critical fault 11b - Multiple Normal-to-Fault-state faults
7-0 NFFS	Normal-to-Fault Freeze Status Used only for testing and debugging. Indicates whether FCCU left the NORMAL state and entered the FAULT state since the last time this register was cleared and, if so, which fault caused FCCU to do so. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00h: No Normal-to-Fault-state transition (cleared) 01h: NCF0 10h: NCF1 ... 7Fh: NCF126 80h: NCF127 ... FFh: Multiple Normal-to-Fault-state transitions

49.14.1.18 Fault-to-Alarm Freeze Status (F2AF_STATUS)

Offset

Register	Offset
F2AF_STATUS	D0h

Function

See F2AF_STATUS[FAFS].

Diagram

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

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

Fields

Field	Function
31-9 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
8-0 FAFS	Fault-to-Alarm Freeze Status Used only for testing and debugging. Indicates whether FCCU left the FAULT state and entered the ALARM state since the last time this register was cleared and, if so, which non-critical fault caused FCCU to do so. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00h: No Fault-to-Alarm-state transition (cleared) 01h: NCF0 10h: NCF1 ... 7Fh: NCF126 80h: NCF127 ... FFh: Multiple Fault-to-Alarm-state transitions

49.14.1.19 Non-critical Fault Fake (NCFF)

Offset

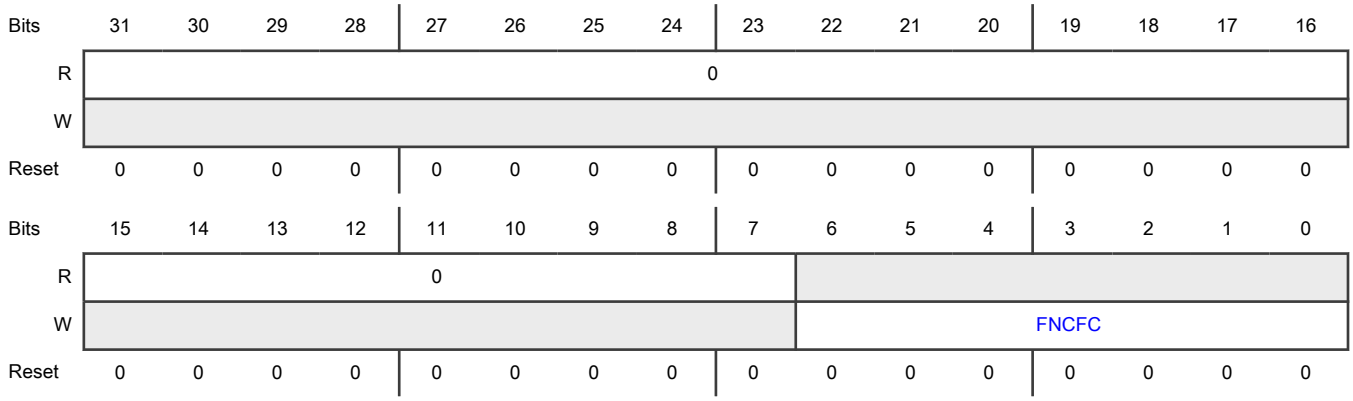
Register	Offset
NCFF	DCh

Function

This register contains a unique code to set a non-critical fault in mutually exclusive mode by the external FAULT interface (signal setting). It allows the SW emulation of the non-critical faults, by injecting the fault directly in the FAULT root, to verify the entire path and reaction. The reaction following a fake non-critical fault cannot be masked.

This is a write-only register with a set of codes corresponding to each non-critical fault injection.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 FNCFC	FNCFC Fake non-critical fault code **NOTE** Writing to this field injects fake faults; writing 00 and the default value being 0 renders different results. 00h: Fake non-critical fault injection at non-critical fault source 0 01h: Fake non-critical fault injection at non-critical fault source 1 02h: Fake non-critical fault injection at non-critical fault source 2 ... 7Fh: Fake non-critical fault injection at non-critical fault source 127

49.14.1.20 IRQ Status (IRQ_STAT)

Offset

Register	Offset
IRQ_STAT	E0h

Function

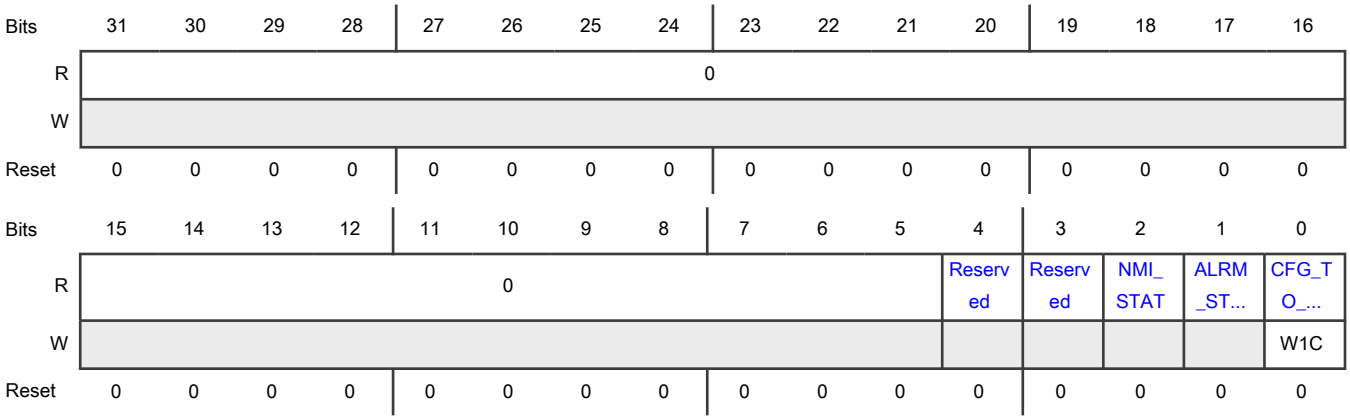
This register provides the FCCU interrupt status related to the following events:

- Configuration-state timeout error
- Alarm interrupt
- NMI interrupt

The configuration-state timeout interrupt is asserted if both IRQ_STAT[CFG_TO_STAT] and IRQ_EN[CFG_TO_IEN] bits are asserted. It is cleared when a 1 is written to the IRQ_STAT[CFG_TO_STAT] bit.

The NMI and ALARM interrupts are asserted and cleared according to the FCCU state. The status bits of the IRQ_STAT trace the status of the related interrupt lines.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 —	Reserved
3 —	Reserved
2 NMI_STAT	NMI Interrupt Status 0b - NMI interrupt is OFF 1b - NMI interrupt is ON
1 ALRM_STAT	Alarm Interrupt Status 0b - Alarm interrupt is OFF 1b - Alarm interrupt is ON
0 CFG_TO_STAT	Configuration-State Timeout Status 0b - No configuration-stat timeout error 1b - Configuration-state timeout error

49.14.1.21 IRQ Enable (IRQ_EN)

Offset

Register	Offset
IRQ_EN	E4h

Function

This register is used to configure enabling of interrupt related to the "Configuration-state timeout error".

The configuration-state timeout interrupt is asserted if both the IRQ_STAT[CFG_TO_STAT] and IRQ_EN[CFG_TO_IEN] fields are set to 1.

Diagram

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

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

Fields

Field	Function
31-4 —	Reserved
3 —	Reserved
2-1 —	Reserved
0 CFG_TO_IEN	Configuration-State Timeout Interrupt Enable 0b - Configuration-state timeout interrupt disabled 1b - Configuration-state timeout interrupt enabled

49.14.1.22 Transient Configuration Lock (TRANS_LOCK)

Offset

Register	Offset
TRANS_LOCK	F0h

Function

See TRANS_LOCK[TRANSKEY]

Diagram

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

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

Fields

Field	Function
31-9 —	Reserved
8-0 TRANSKEY	Transient Configuration Lock Writable only by code running in Supervisor mode. Temporarily locks and unlocks the configuration. Locking the configuration prevents FCCU from entering the CONFIG state. For information about putting FCCU in configuration, see Prepare FCCU for configuration and Configure FCCU. **NOTE** You can write to this field when FCCU is in any state, but the lock will not get into effect until FCCU is in the NORMAL state. BCh: Unlock. Any other value: Lock.

49.14.1.23 Permanent Configuration Lock (PERMNT_LOCK)

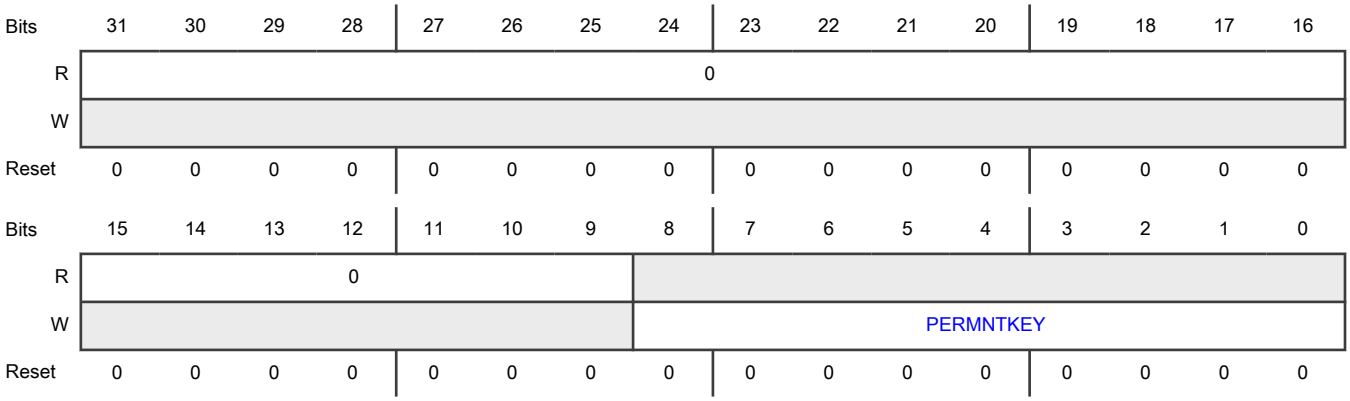
Offset

Register	Offset
PERMNT_LOCK	F4h

Function

See PERMNT_LOCK[PERMNTKEY]

Diagram



Fields

Field	Function
31-9 —	Reserved
8-0 PERMNTKEY	Permanent Configuration Lock Writable only by code running in the Supervisor mode. Permanently locks the configuration, which prevents FCCU from entering the CONFIG state until FCCU is reset. For information about putting FCCU in configuration, see Prepare FCCU for configuration and Configure FCCU. **NOTE** You can write to this field when FCCU is in any state, but the lock will not get into effect until FCCU is in NORMAL state. FFh: Lock. Any other value: Do nothing.

49.14.1.24 Delta T (DELTA_T)

Offset

Register	Offset
DELTA_T	F8h

Function

The DELTA_T register is used for programming the value of delta_T constant (see DELTA_T), in microseconds.

NOTE

This register can be written only when the FCCU is in the CONFIG state.

Diagram

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

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

Fields

Field	Function
31-30 —	Reserved
29-16 —	Reserved
15-14 —	Reserved
13-0 DELTA_T	Minimum Fault-Output (EOUT) Timer Interval Applies only to Bi-Stable mode (CFG[FOM]). Controls the minimum amount of time (T_{min}) that the fault-output (EOUT) timer runs according to this equation: T_{min} = 250 µs + DELTA_T

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The durations shown for the DELTA_T values are for CLKSAFE clock signals running at exactly 48 MHz. However, the CLKSAFE signals are sourced from an internal chip clock signal (for example, IRC)—or an integer division of it (see the chip-specific FCCU information)—that has a trimmed frequency variation. Therefore, the DELTA_T durations may vary. See your chip's data sheet for the trimmed frequency variation (for example, δF_{var}).

49.14.1.25 Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0)

Offset

Register	Offset
IRQ_ALARM_EN0	FCh

Function

See IRQ_ALARM_ENa[IRQENn].

Diagram

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

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

Fields

Field	Function
31-8 —	Reserved
7-0 IRQENn	Non-critical Alarm-State Interrupt-Request Enable n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the interrupt request is enabled as the Alarm-state

Table continues on the next page...

Table continued from the previous page...

Field	Function
	reaction for the associated non-critical fault channel (n). When the ALARM state and the Alarm-state interrupt request are enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to assert the irq_alarm signal when FCCU enters the ALARM state; irq_alarm remains asserted until FCCU is in the NORMAL state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. 0b - Disabled 1b - Enabled

49.14.1.26 Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0)

Offset

Register	Offset
NMI_EN0	10Ch

Function

See NMI_ENa[NMIENn].

Diagram

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

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

Fields

Field	Function
31-8 —	Reserved
7-0 NMIENn	Non-critical Fault-State Non-maskable-Interrupt-Request Enable n

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the non-maskable interrupt request is enabled as a Fault-state reaction for the associated non-critical fault channel (n). When the non-maskable interrupt request is enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to assert the NMIOUT signal when FCCU enters the FAULT state; NMIOUT remains asserted until FCCU exits FAULT state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. 0b - Disabled 1b - Enabled

49.14.1.27 Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0)

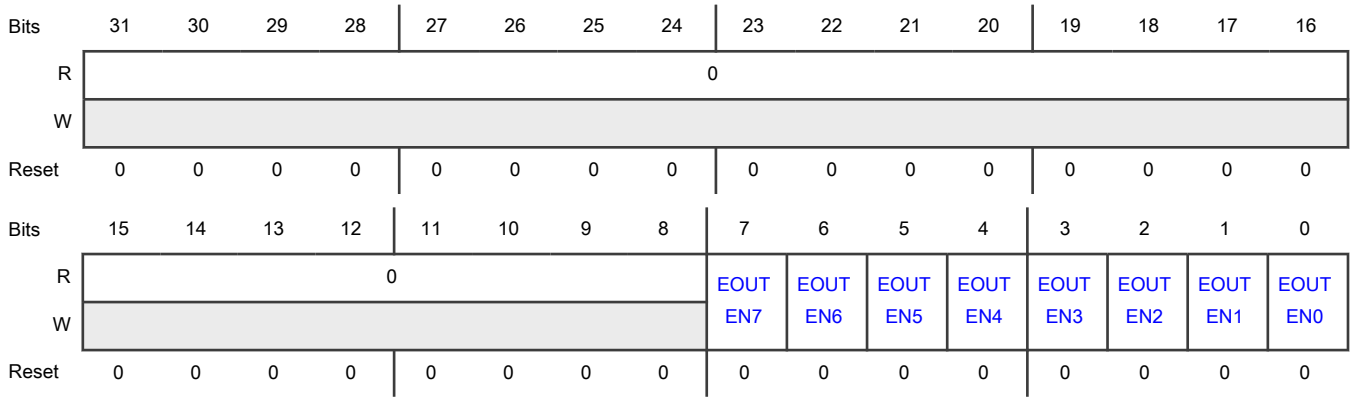
Offset

Register	Offset
EOUT_SIG_EN0	11Ch

Function

See EOUT_SIG_ENa[EOUTENn].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0	Non-critical Fault-State EOUT Signaling Enable n

Table continues on the next page...

Table continued from the previous page...

Field	Function
EOUTENn	Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Applies only when the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]). When FCCU is configured for Bi-Stable fault-output mode (CFG[FOM]), controls whether EOUT signaling is enabled as a Fault-state reaction for the associated non-critical fault channel (n). (For other fault-output modes, EOUT signaling is always enabled, regardless of the value of this field.) When EOUT signaling is enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to indicate the faulty condition on the EOUT[1:0] signals when FCCU enters the FAULT state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. For all fault-output modes, also controls whether FCCU asserts the FIF signal when a fault on the associated non-critical fault channel (n) causes FCCU to enter the FAULT state. **NOTE** For all fault-output modes, you must set this field to enabled to ensure that FCCU asserts the FIF signal when FCCU enters FAULT state as the result of a fault on the associated non-critical fault channel (n) so the FOSU module does not mistakenly generate a destructive chip reset. 0b - In Bi-Stable fault-output mode, both EOUT signaling and FIF assertion are disabled; in other fault-output modes, EOUT signaling is enabled and FIF assertion is disabled. 1b - Both EOUT signaling and FIF assertion are enabled in all fault-output modes.

49.14.1.28 Alarm-State Timer (TMR_ALARM)

Offset

Register	Offset
TMR_ALARM	12Ch

Function

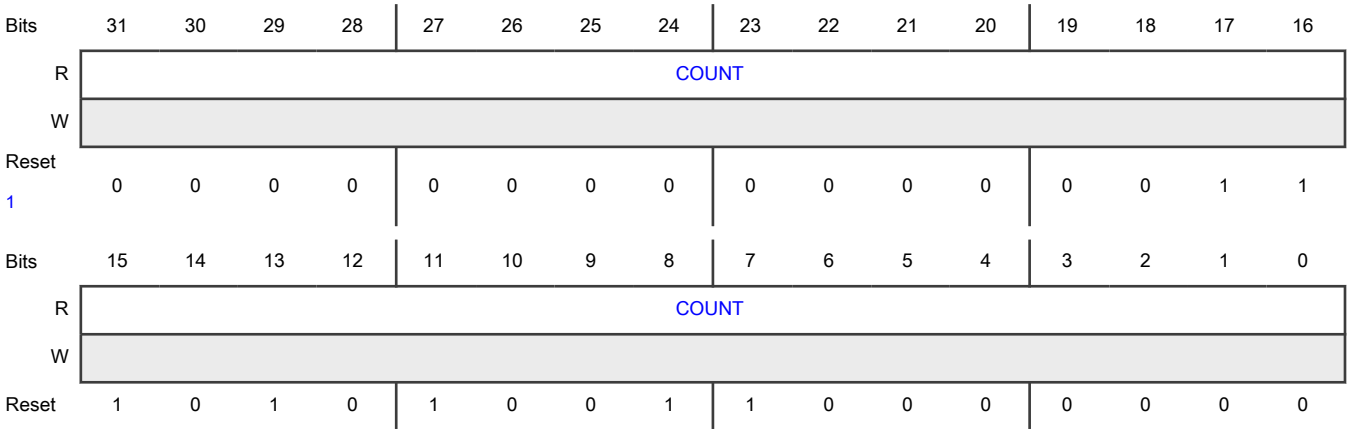
See TMR_ALARM[COUNT].

This table shows how the Alarm-state timer's state and value vary by FCCU state:

Table 235. TMR_ALARM reset value

FCCU state	Timer state	Timer value
CONFIG	Idle	0000_0000h
NORMAL	Idle	Initial value: NCF_TO[TO]
ALARM	Running	Value when read
FAULT	Idle	End of count

Diagram



1. The default reset value is provided by NCF_TO[TO]. See Table 235 for the reset value at different FCCU states.

Fields

Field	Function
31-0 COUNT	Alarm-State Timer Count Specifies the value of the Alarm-state timer in CLKSAFE periods.

49.14.1.29 Configuration-State Timer (TMR_CFG)

Offset

Register	Offset
TMR_CFG	134h

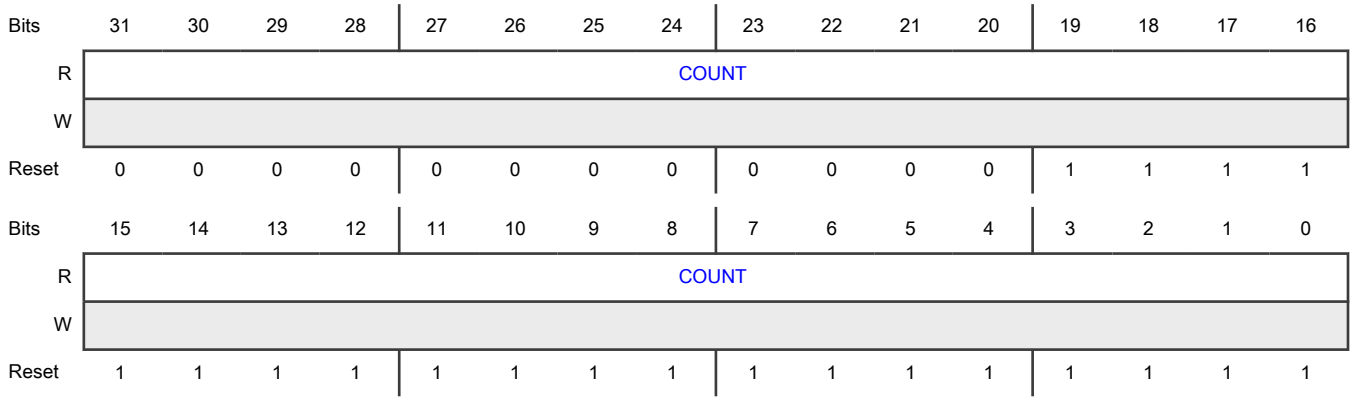
Function

See TMR_CFG[COUNT].

This table shows how the Configuration-state timer's state and value vary by FCCU state:

FCCU state	Timer state	Timer value
CONFIG	Running	Value when read
NORMAL	Idle	000F_FFFFh
ALARM	Idle	000F_FFFFh
FAULT	Idle	000F_FFFFh

Diagram



Fields

Field	Function
31-0 COUNT	Configuration-State Timer Count Specifies the value of the Configuration-state timer in CLKSAFE periods.

49.14.1.30 Fault-Output Timer (TMR_ETMR)

Offset

Register	Offset
TMR_ETMR	138h

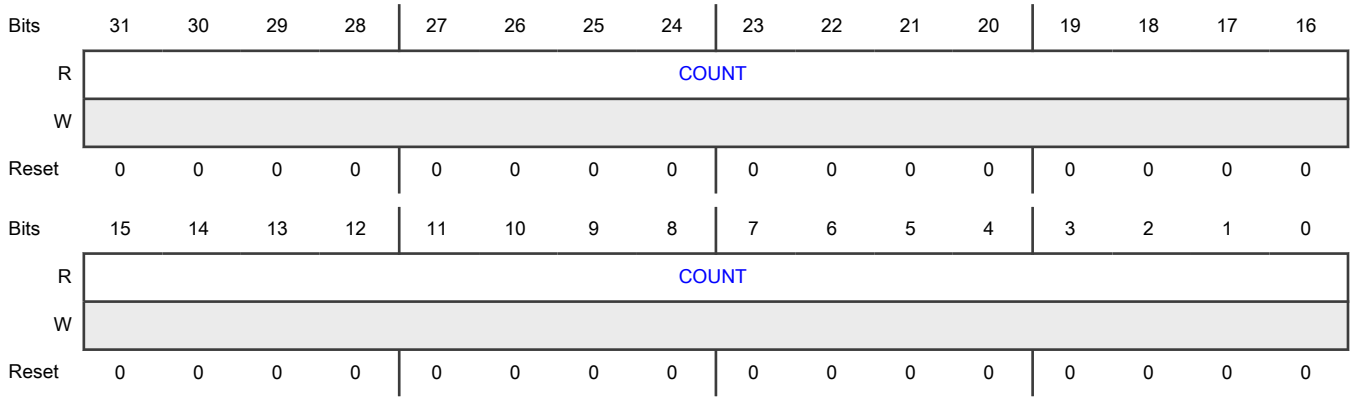
Function

See TMR_ETMR[COUNT].

This table shows how the fault-output timer's state and value vary by FCCU state:

FCCU state	Timer state (value)
CONFIG	Idle (0000_0000h)
NORMAL	Idle (0000_0000h)
ALARM	Idle (0000_0000h)
FAULT	Running (value when read)

Diagram



Fields

Field	Function
31-0 COUNT	Fault-Output Timer Count Specifies the value of the fault-output timer in CLKSAFE periods.

49.14.2 Configuration registers

49.14.2.1 Definition

Configuration registers are registers that:

- Let you configure FCCU's Alarm-state timer interval, fault channels, and fault-output (EOUT) signals.
- You can write to only when the configuration is not locked and FCCU is in CONFIG state (see Put FCCU in the CONFIG state).
- Save the values you write to them while FCCU is in CONFIG state only after you manually put FCCU in NORMAL state. If FCCU automatically leaves CONFIG state and enters NORMAL state because the configuration-timer interval (CFG_TO[TO]) expires (called a Configuration-state timeout), FCCU changes the value of the Configuration (CFG) register to its Configuration-state-timeout value and the value of each of the other configuration registers to its reset value; FCCU also changes the value of the Configuration-State Timeout Interval (CFG_TO) register to its reset value. For information on the Configuration-state timeout value, see CFG register bit value at different events.

49.14.2.2 Configuration registers

Following is the list of configuration registers in this module. They are listed in an offset order from lowest to highest.

- Configuration (CFG)
- Non-critical Fault Configuration (NCF_CFGa)
- Non-critical Fault-State Configuration (NCFS_CFG0)
- Non-critical Fault Enable (NCF_E0)
- Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0)
- Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)
- Delta T (DELTA_T)
- Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0)
- Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0)

- Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0)

49.14.3 CFG register bit value at different events

In this chip, there are no events that affect the bits in the configuration register.

49.15 Glossary

CF	Critical fault
EOUT	Error out
FOSU	FCCU output supervision unit
FSM	Finite state machine
intf	Interface
IRQ	Interrupt request
NCF	Non-critical fault
NMI	Non-maskable interrupt

Chapter 50
Self-test General-Purpose Registers (SELFTEST_GPR)

50.1 Introduction

SELFTEST_GPR configures the STCU2 LBIST shift cycles. Programming self-test GPRs is a part of the self-test programming sequence. See the "Self-test programming sequence" section in the STCU2 chapter.

NXP supports only STCU2 Built-In Self-Test (BIST) sequences that have been validated by NXP for in-field testing usage.

The detailed programming sequences for supported configurations are available in a separate application note (contact your NXP sales representative).

The values for SELFTEST_GPR registers must be aligned with the NXP supported configurations only.

NOTE

SELFTEST_GPR is not present in MWCT2016S and MWCT2015S.

NOTE

While accessing SELFTEST_GPR space, software must ensure that the STCU2 block has its clock enabled, i.e., MC_ME.PRTN1_COFB3_CLKEN[REQ104] configured as 1'b1. Not ensuring this might result in unpredictable device behaviour.

50.2 Peripheral shift clock switching (PCS)

The PCS feature is used to avoid current surges at the start and/or end of LBIST. Since LBIST can cause sudden current surges which might lead to chip malfunction.

With PCS enabled, there are additional (SELFTEST_GPR.CONFIG_GPR[PCS_STEP_SIZE] + 1)*16 patterns beyond the configured count. If enabled at the start of LBIST (using SELFTEST_GPR.CONFIG_GPR[PCS_ENABLE_START]), it would cause the shift clock to scale up in this duration from divide by 16 to the original frequency (by changing divider in each step).

Similarly, if configured at the end of LBIST (using SELFTEST_GPR.CONFIG_GPR[PCS_ENABLE_END]), the shift clock frequency will be scaled down from original frequency to divide by 16.

50.3 SELFTEST_GPR register descriptions

50.3.1 SELFTEST_GPR memory map

SELFTEST_GPR base address: 403B_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Configuration register (CONFIG_REG)	32	RW	0000_ADD5h
14h	LBIST Program (LBIST_PROG_REG)	32	RW	0003_0050h

50.3.2 Configuration register (CONFIG_REG)

Offset

Register	Offset
CONFIG_REG	0h

Diagram

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

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

Fields

Field	Function
31 —	Reserved
30-17 —	Reserved
16 —	Reserved
15-9 —	Reserved
8 PCS_ENABLE_END	PCS Enable End Enables Progressive Shift Clock Switching for LBIST at end. The feature gets enabled when this bit is set to 1. It is recommended to run LBIST patterns by enabling this feature.
7 PCS_ENABLE_START	PCS Enable Start Enables Progressive Shift Clock Switching for LBIST at start. The feature gets enabled when this bit is set to 1. It is recommended to run LBIST patterns by enabling this feature.
6-4	PCS Step Size pcs_step_size[2:0] controls the step size for progressive increase of shift frequency:

Table continues on the next page...

Table continued from the previous page...

Field	Function
PCS_STEP_SIZE	000 - step size of 1 pattern 001 - step size of 2 patterns ... 111 - step size of 8 patterns
3-0 —	Reserved

50.3.3 LBIST Program (LBIST_PROG_REG)

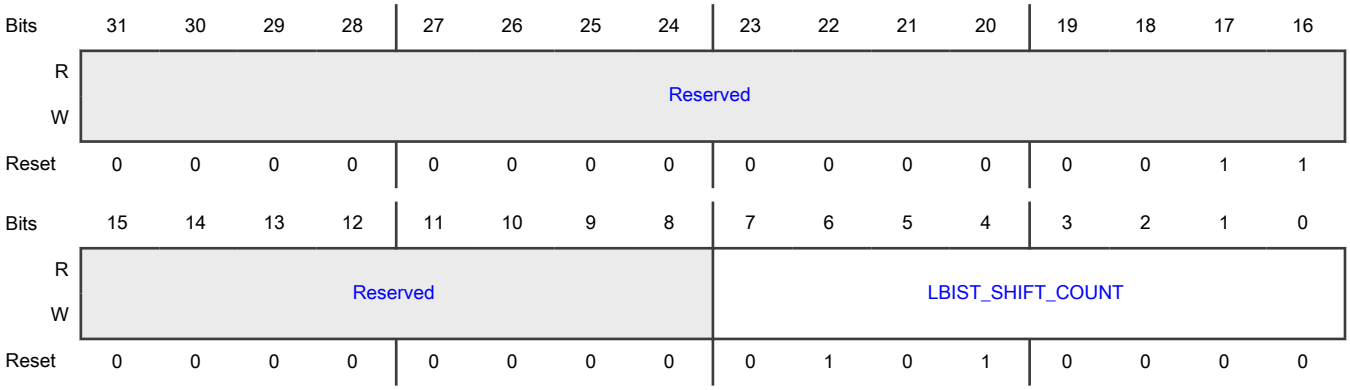
Offset

Register	Offset
LBIST_PROG_REG	14h

Function

Configures the LBIST shift count.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 LBIST_SHIFT_COUNT	LBIST Shift Count Specifies the number of shift cycles in a scan chain in one pattern count for the LBIST partition.

50.4 Glossary

BIST clock	The BIST clock refers to the BIST operational clock which is the clock source used while LBIST is operational by the LBIST controller, which is the MC_CGM clock node LBIST_CLK for the chip. See the system block diagram for details.
LBIST	Logic built-in self-test
Pattern count	The pattern count refers to an empirically determined value of shift patterns run in LBIST on a scan chain to result in predetermined fault coverage, based on device safety integrity level. Since the LBIST scheme is Pseudo Random Testing, a sufficiently good value of pattern count is used which provides the required fault coverage.
Scan chain	A sequential chain of flops in scan mode over which one scan pattern (generated by PRPG, that is, Pseudo Random Pattern Generator) in LBIST will be shifted and the output MISR (Multi Input Signature Register) will be matched with earlier known value to determine pass or fail criterion of that particular chain.

Chapter 51
Self-Test Control Unit (STCU2)

51.1 Chip-specific STCU2 information

51.1.1 Supported BIST sequences

This chip supports only STCU2 BIST sequences that NXP has validated for in-field testing usage. The detailed programming sequences for supported configurations are available in a separate application note (contact your NXP sales representative for this information).

51.1.2 Self-test overview

STCU2 provides the overall control of both LBIST and MBIST, in serial or parallel, depending on the power, timing, or coverage constraints.

NOTE

LBIST is not available for MWCT2016S and MWCT2015S.

The application software initiates the self-test sequence or procedure (please use whichever term is correct), and has provisions to bypass the self-test configuration to directly move to normal Run mode.

There is a programmable BIST-sequence-execution watchdog timer that specifies the maximum time allowed for the execution of a BIST sequence. In case the selected LBISTs or MBISTs are not complete during the assigned time, the current LBISTs or MBISTs execution is interrupted and a failure is flagged into ERR_STAT[WDTOSW] and MBESWx or LBESW.

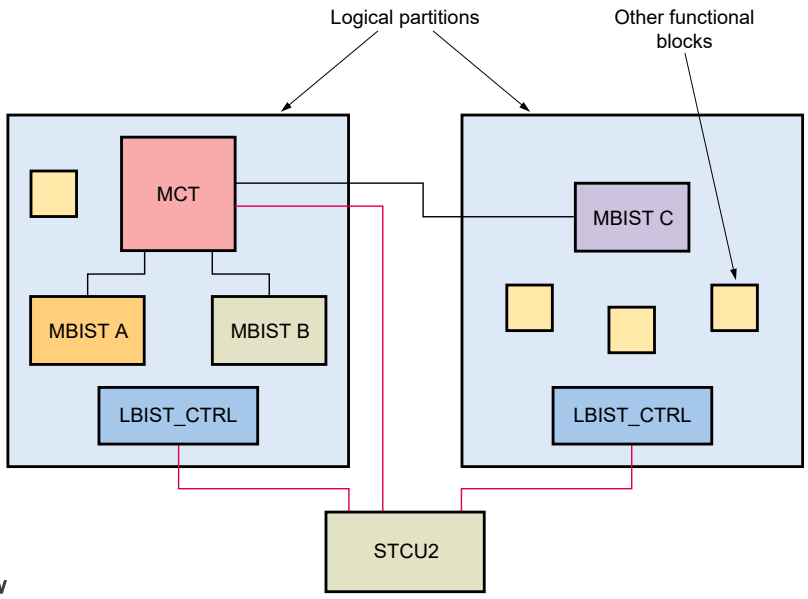
The STCU2 execution includes three phases:

- Software configuration. Load or program the test configuration parameters to be executed, depending on ASIL target and performance requirements. The self-test configuration parameters allow the self-test sequence to run with different times depending on pattern numbers and types, achieving different levels of coverage.
- Start of BIST execution. This involves software register configuration in STCU2. The BIST completion is followed by a functional reset.
- End of BIST execution. After BIST execution and functional reset are complete, the BIST results are made available and you decide how to continue (trigger a new reset or start a safety application). The BIST execution completion is signalized in an internal STCU2 register (LBESWn) bit which is not affected by the functional reset. The BIST test results are also not affected by the functional reset.

The following figure shows a pictorial representation of two LBIST partitions having two and one MBIST groups, respectively. STCU2 interacts with LBIST_CTRL for the respective partitions to configure and control the LBIST sequence and compare results.

It also interacts with MTR controller (MCT) for MBIST. Each LBIST partition has its corresponding controller whereas a single MCT can cater to multiple MBIST groups.

Figure 191.



Self-test overview

NOTE

LBIST_CTRL is not available for MWCT2016S and MWCT2015S.

Each LBIST partition also includes a block called Self-test GPR involving a set of registers that provide miscellaneous configuration and parameter details for an LBIST operation.

When you trigger a self-test sequence, the chip becomes unavailable for software application during the self-test run.

51.1.3 STCU2 LBIST/MBIST mapping

This topic covers STCU2 LBIST/MBIST mapping.

LBIST mapping

The below table lists the LBIST mapping of the chip. The NLBIST value is 1.

Table 236. LBIST mapping

BIST ID (NLBIST)	BIST instance name	Modules
0	LBIST	DMA: • DMA controller
		EIM
		ERM
		XRDC • MDAC0 • MDAC1

Table continues on the next page...

Table 236. LBIST mapping (continued)

BIST ID (NLBIST)	BIST instance name	Modules
		• MDAC2 • MDAC3 • MDAC4 • MDAC5 • MRC0 • MRC1 • MRC2 • PDAC0 • PDAC1 • PDAC2
		CRC
		FCCU
		FOSU
		PFLASH_CTL
		PRAMC_0
		PRAMC_1 **NOTE** PRAMC_1 is not present in MWCT2016S, and MWCT2015S.
		SWT_1 **NOTE** SWT_1 is not present in MWCT2016S and MWCT2015S.
		CMU: • CMU1 • CMU2 • CMU4 • CMU5
		iAHB Gaskets: • DMA_GSKT • HSE_GSKT • AIPS1_GSKT • AIPS2_GSKT

Table continues on the next page...

Table 236. LBIST mapping (continued)

BIST ID (NLBIST)	BIST instance name	Modules
		• QSPI_GSKT **NOTE** AIPS2_GSKT and QSPI_GSKT are not present in MWCT2016S and MWCT2015S. • TCM_GSKT • EMAC_GSKT • EDC master checker gaskets (See block diagram in Introduction chapter for details) • EDC slave checker gasket (See block diagram in Introduction chapter for details)
		Crossbars • AXBS • DMA_AXBS • HSE_AXBS • PERIPHERAL_AXBS • TCM_AXBS

MBIST mapping

The below tables lists the MBIST mapping of different variants of this chip.

Table 237. MWCT2D17S MBIST mapping

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
0	SYS0_RAMS	0	PRAM0 block 1
0	SYS0_RAMS	1	PRAM0 block 2
1	SYS1_RAMS	0	PRAM1 block 1
1	SYS1_RAMS	1	PRAM1 block 2
2	DMA_TCD_RAM	0	DMA TCD RAM
3	CM7_0_TOP	0	CM7_0 instruction cache data block 1
3	CM7_0_TOP	1	CM7_0 instruction cache data block 2
3	CM7_0_TOP	2	CM7_0 instruction cache tag block 1
3	CM7_0_TOP	3	CM7_0 instruction cache tag block 2
3	CM7_0_TOP	4	CM7_0 data cache data block 1
3	CM7_0_TOP	4	CM7_0 data cache data block 2
3	CM7_0_TOP	5	CM7_0 data cache data block 3
3	CM7_0_TOP	5	CM7_0 data cache data block 4

Table continues on the next page...

Table 237. MWCT2D17S MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
3	CM7_0_TOP	6	CM7_0 data cache data block 5
3	CM7_0_TOP	6	CM7_0 data cache data block 6
3	CM7_0_TOP	7	CM7_0 data cache data block 7
3	CM7_0_TOP	7	CM7_0 data cache data block 8
3	CM7_0_TOP	8	CM7_0 data cache tag block 1
3	CM7_0_TOP	9	CM7_0 data cache tag block 2
3	CM7_0_TOP	10	CM7_0 data cache tag block 3
3	CM7_0_TOP	11	CM7_0 data cache tag block 4
3	CM7_0_TOP	12	CM7_0 instruction TCM (Tightly Coupled Memory)
3	CM7_0_TOP	13	CM7_0 data TCM block 1
3	CM7_0_TOP	13	CM7_0 data TCM block 2
4	CM7_1_TOP	0	CM7_1 instruction cache data block 1
4	CM7_1_TOP	1	CM7_1 instruction cache data block 2
4	CM7_1_TOP	2	CM7_1 instruction cache tag block 1
4	CM7_1_TOP	3	CM7_1 instruction cache tag block 2
4	CM7_1_TOP	4	CM7_1 data cache data block 1
4	CM7_1_TOP	4	CM7_1 data cache data block 2
4	CM7_1_TOP	5	CM7_1 data cache data block 3
4	CM7_1_TOP	5	CM7_1 data cache data block 4
4	CM7_1_TOP	6	CM7_1 data cache data block 5
4	CM7_1_TOP	6	CM7_1 data cache data block 6
4	CM7_1_TOP	7	CM7_1 data cache data block 7
4	CM7_1_TOP	7	CM7_1 data cache data block 8
4	CM7_1_TOP	8	CM7_0 data cache tag block 1
4	CM7_1_TOP	9	CM7_0 data cache tag block 2
4	CM7_1_TOP	10	CM7_0 data cache tag block 3
4	CM7_1_TOP	11	CM7_0 data cache tag block 4
4	CM7_1_TOP	12	CM7_0 instruction TCM
4	CM7_1_TOP	13	CM7_0 data TCM block 1
4	CM7_1_TOP	13	CM7_0 data TCM block 2
5	FLEX_CAN_RAMS	0	FLEXCAN0 MB memory
5	FLEX_CAN_RAMS	1	FLEXCAN1 MB memory

Table continues on the next page...

Table 237. MWCT2D17S MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
5	FLEX_CAN_RAMS	2	FLEXCAN2 MB memory
5	FLEX_CAN_RAMS	3	FLEXCAN3 MB memory
5	FLEX_CAN_RAMS	4	FLEXCAN4 MB memory
5	FLEX_CAN_RAMS	5	FLEXCAN5 MB memory
6	QSPI_PERI_RAMS	0	QSPI RAM
6	QSPI_PERI_RAMS	1	QSPI TX memory
7	EMAC_TSN_RAM	0	EMAC Timestamp Memory
8	EMAC_RAMS	0	EMAC RXParser
8	EMAC_RAMS	1	EMAC TX block 1
8	EMAC_RAMS	2	EMAC TX block 1
8	EMAC_RAMS	3	EMAC TX block 2
8	EMAC_RAMS	4	EMAC RX block 2
9	b03_ETF_RAMS	0	HTM ETF
9	b03_ETF_RAMS	1	Shared ETF
9	b03_ETF_RAMS	2	CM7 cluster Instruction ETF
9	b03_ETF_RAMS	3	CM7 cluster Data ETF
10	HSE_RAMS	0	HSE secure RAM
10	HSE_RAMS	1	HSE PKC memory block 1
10	HSE_RAMS	2	HSE PKC memory block 2
10	HSE_RAMS	3	HSE MTB memory
11	HSE_ROMS	0	BOOTROM block 1
11	HSE_ROMS	0	BOOTROM block 2
11	HSE_ROMS	0	BOOTROM block 3
11	HSE_ROMS	0	BOOTROM block 4
11	HSE_ROMS	0	BOOTROM block 5

Table 238. MWCT2016S MBIST mapping

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
0	HSE_ROMS	0	BOOTROM block 1
0	HSE_ROMS	0	BOOTROM block 2
0	HSE_ROMS	0	BOOTROM block 3
0	HSE_ROMS	0	BOOTROM block 4

Table continues on the next page...

Table 238. MWCT2016S MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
0	HSE_ROMS	0	BOOTROM block 5
1	HSE_RAMS	0	HSE secure RAM
1	HSE_RAMS	1	HSE PKC memory block 1
1	HSE_RAMS	2	HSE PKC memory block 2
1	HSE_RAMS	3	HSE MTB memory
2	SYS0_DMA_RAMS	0	PRAM0 block 1
2	SYS0_DMA_RAMS	1	PRAM0 block 2
2	SYS0_DMA_RAMS	2	DMA TCD RAM
3	CM7_0_TOP	0	CM7_0 instruction cache data block 1
3	CM7_0_TOP	1	CM7_0 instruction cache data block 2
3	CM7_0_TOP	2	CM7_0 instruction cache tag block 1
3	CM7_0_TOP	3	CM7_0 instruction cache tag block 2
3	CM7_0_TOP	4	CM7_0 data cache data block 1
3	CM7_0_TOP	4	CM7_0 data cache data block 2
3	CM7_0_TOP	5	CM7_0 data cache data block 3
3	CM7_0_TOP	5	CM7_0 data cache data block 4
3	CM7_0_TOP	6	CM7_0 data cache data block 5
3	CM7_0_TOP	6	CM7_0 data cache data block 6
3	CM7_0_TOP	7	CM7_0 data cache data block 7
3	CM7_0_TOP	7	CM7_0 data cache data block 8
3	CM7_0_TOP	8	CM7_0 data cache tag block 1
3	CM7_0_TOP	9	CM7_0 data cache tag block 2
3	CM7_0_TOP	10	CM7_0 data cache tag block 3
3	CM7_0_TOP	11	CM7_0 data cache tag block 4
3	CM7_0_TOP	12	CM7_0 instruction TCM (Tightly Coupled Memory)
3	CM7_0_TOP	13	CM7_0 data TCM block 1
3	CM7_0_TOP	13	CM7_0 data TCM block 2
4	FLEX_CAN_RAMS	0	FLEXCAN0 MB memory
4	FLEX_CAN_RAMS	1	FLEXCAN1 MB memory
4	FLEX_CAN_RAMS	2	FLEXCAN2 MB memory
4	FLEX_CAN_RAMS	3	FLEXCAN3 MB memory
4	FLEX_CAN_RAMS	4	FLEXCAN4 MB memory

Table continues on the next page...

Table 238. MWCT2016S MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
4	FLEX_CAN_RAMS	5	FLEXCAN5 MB memory

There are different number of MBIST indices (NMCUT) across different parts. The below table shows the variation across different chips.

Table 239. NMCUT

Chip	NMCUT
MWCT2D17S	12
MWCT2016S	5

51.1.4 Clock connectivity

STCU2 works on the system clock domain. The scan operations while LBIST are done on TCK clock.

Table 240. Clock connectivity

STCU2 clock	Clock connected to the chip	Remarks
STCU2 CORE_CLK	AIPS_SLOW_CLK	System clock divider output clock node for STCU2 operations and register interface accesses
REG_INTF_CLK		
SHIFT_CLK	JTAG_TCK	JTAG_TCK used for shift operations when self-test is in progress

51.1.5 Memory map for self-test GPRs

See the memory map file attached to this document and 'SELFTEST_GPR' chapter for details.

51.1.6 Handling unrecoverable faults

Only recoverable and non-critical faults from STCU2 are mapped to FCCU. The unrecoverable and critical faults are mapped to MC_RGM. Therefore, ignore any references to mapping of unrecoverable and critical faults with FCCU in this chapter.

51.1.7 Indicating fault state during the main reset domain Self-Test

FCCU EOUT indicates the fault state during the Self-Test as soon as the UTEST_MISC[FCCU_EOUT_ DEDICATED] is enabled and DCMRWD2[EOUT_STAT_DUR_STEST] is enabled at the time of Self-Test setup steps.

The fault state remains asserted until the bit DCMRWD2[EOUT_STAT_DUR_STEST] is cleared post Self-Test reset sequence.

In case it is not required to indicate the fault state during the Self-Test, then the bit DCMRWD2[EOUT_STAT_DUR_STEST] should not be programmed.

51.1.8 Error events management in main reset domain Self-Test

Error events should be mapped to unrecoverable faults in STCU2_ERR_FM register for main reset domain Self-Test.

51.1.9 AUTOLOCK_VALUE for register write access via STCU2_SKC

Register write-access watchdog timer explains that access to STCU2 registers via the STCU2_SKC security-key mechanism times out after a number of STCU2 clock cycles equal to AUTOLOCK_VALUE. On this chip, AUTOLOCK_VALUE is 32'hFFFF_FFFF.

51.1.10 PLL loss of lock during Self-Test

Detection of PLL loss of lock during LBIST or MBIST causes a destructive reset in the system.

CMU0 programming is required before the Self-Test programming begins.

All the registers present in LBIST domains do not retain their content after Self-Test reset.

51.1.11 STCU2 BIST start (BSTART) register description

Attempted accesses with the undefined reserved values in the BSTART register can result in undefined behavior. The following table defines the BSTART field on this chip.

Table 241. BSTART description

Field value	Description
000b	NOP (reset value)
001b	RUN_ONLY: Run BISTs. MTR controller (MCT) runs the selected BISTs without programming them first.
010b	Reserved
011b	Reserved
100b	PROG_ONLY: Program BISTs only. MCT only programs BISTs and does not start them. This enables different BISTs to be programmed with different algorithms before all the selected BISTs are started using the RUN_ONLY command.
101b	PROG_RUN: Program BISTs and start them.
110b	Reserved
111b	Reserved

51.1.12 STCU2 algorithm select (ALGOSEL) register description

This register selects any one of the predefined algorithms. The value used in this register is mapped to a predefined algorithm register in BIST. After MCT has programmed information on this register and starts some BIST operation, this value is transferred to the predefined algorithm register in every BIST that is selected.

If multiple predefined algorithms are selected, the order of execution is from LSB to MSB.

Any field that does not have a predefined algorithm associated with it is a reserved bit. No data is written to reserved bits and when you read these bits, they always return 0. These predefined algorithms include:

- Backgrounds
- Address modes and other register programming that may be necessary
- The number of BIST runs needed to perform the exact coverage requested.

MCT decodes this information through a series of look-up tables. The following table describes the high-level BIST algorithms that can be invoked through the MCT. The columns in that table have the following meaning:

- Index: The position in this MCT register enables this algorithm.
- Name: The symbolic name for this algorithm.
- Description: The intent and use model for this algorithm.
- BIST runs: The sequence of BIST invocations comprising this algorithm.
- Sequences: The ordered list of march elements to run. Each march element consists of a direction indicator (for increasing addresses and for decreasing addresses) and a set of march phases. A march phase consists of reading (R) or writing (W), the non-inverted (0) or inverted (1) background pattern.
- Used by default: Whether this algorithm is selected in the post-reset state of this register.

Table 242. ALGOSEL description

Index	Name	Description	BIST runs	Sequences	Used by default
3	March C+ single	March C+ with column fast and solid background	MarchC+ column fast 1x (#4)	W0, R0W1, R1W0R0W1, R1W0, R0W1R1W0, R0	Yes
13	BasicChk	Basic selfcontained checkerboard, using column fast	BasicChk (#18)	R0W1R1	-

51.1.13 Self-Test programming sequence

This section describes a high level programming sequence for executing Self-Test. The detailed programming sequences for supported configurations are available in a separate application note (contact your NXP sales representative).

- Program clock sources and MC_CGMs as per the clock configuration.

  NOTE

  While running MBIST on EMAC timestamp memory, MC_CGM.MUX_9_DC_0[DIV] should be appropriately configured to ensure EMAC_CLK_TS should be atleast 1.5 times the AIPS_SLOW_CLK frequency.

- Program SELFTEST_GPR[CONFIG_REG] and SELFTEST_GPR[LBIST_PROG_REG] as per the LBIST configurations.

  NOTE

  This is not applicable for MWCT2016S and MWCT2015S wherein LBIST is not supported.

- Program STCU2 as per NXP recommended.
- Program dcf_client_utest_misc[FCCU_EOUT_DEDICATED] and DCMRWD2[EOUT_STAT_DUR_STEST] to configure EOUT pins as dedicated and to indicate FCCU error states on EOUT pins during Self-Test respectively.
- Configure ERCTRL[ERASSERT] register bit in MC_RGM to achieve desired behaviour of Reset Pin as documented in Reset Section and Pad States Section.
- Disable functional reset sources to avoid false Self-Test abort scenarios. For example, software watchdog timers cannot be serviced while Self-Test because the software is not accessible and can cause invalid timeout reset. See Reset events and configurations while Self-Test.
- Program the PCS_ENABLE[8:7] fields of the CONFIG_REG register (SELFTEST_GPR) to 0 to disable PCS.
- Trigger Self-Test execution by programming RUNSW register in STCU2.
- At the end of the Self-Test run, a functional reset is generated. Out of reset, the software can check the status registers inside STCU2 and take actions accordingly.

51.1.14 Modules operation during/post Self-Test

The clock, reset, power generation modules, such as RGM, PMC, CGM, FIRC, PLL, and SIRC will be available during and post Self-Test. However, because the chip undergoes the reset sequence, PLL and CGM will be reset.

CMU0 and CMU3 will be available during Self-Test if enabled prior to Self-Test, but will be Reset post Self-Test.

STCU2 will also be available during Self-Test and the results can be read post Self-Test.

51.1.15 Reset events and configurations while Self-Test

As Self-Test is a software initiated Self-Test, it is expected by software to ensure all the peripherals are disabled as documented in Software considerations before Self-Test.

The functional reset sources should be appropriately configured to avoid any functional reset while Self-Test is running (disable SWT_RST and HSE_SWT_RST. JTAGC should not be in EXTEST, HIGHZ or CLAMP instructions). This ensures no selftest abort while running selftest due to functional reset source. The below table specifies the state of functional reset sources for Self-Test.

Table 243. Reset events and configurations while self-test

Reset source	Reset description	Configurations/remarks for selftest
FCCU_RST	FCCU reset reaction	The FCCU is a part of logic which is under Self-Test and thus it would be disabled during the Self-Test
ST_DONE	Self-Test done reset	The selftest done indication resets the device post selftest. While Self-Test is running, the Self-Test done indication will be in disabled state
SWT_RST	SWT reset request	SWT should be disabled in Self-Test configuration sequence.
JTAG_RST	JTAG Reset	JTAG communication should be inactive while Self-Test and JTAGC state machine should not be in EXTEST, HIGHZ or CLAMP instructions
HSE_SWT_RST	HSE SWT timeout	HSE_SWT should be disabled in Self-Test configuration sequence.
SW_FUNC	software 'functional' reset	The software is inactive during Self-Test and hence software functional reset will not arrive while the chip is under Self-Test.
DEBUG_FUNC	debug 'functional' reset	MDM DAP should not be configured for debug functional reset generation while Self-Test is running.

The destructive reset sources are available while in Self-Test. In case of any destructive reset event, the device undergoes a destructive reset sequence, thereby resetting the STCU2 and Self-Test logic itself.

After completion of Self-Test (LBIST or MBIST), a functional reset sequence is executed by MC_RGM.

51.1.16 Software considerations before Self-Test

1. As all the communication peripherals are not in LBIST partitions; hence, it is required by SW to stop all communications and disable all the PCTLs of the communication peripherals before initiating Self-Test as the device will be unusable in Self-Test. This is needed so that there is graceful safe stating of inputs of non-lbisted peripheral modules.
2. Configure RGM functional reset sources as interrupts before Self-Test as per the considerations mentioned in Reset events and configurations while Self-Test, i.e., the SWTs should be disabled, the JTAGC should not be in states which can cause a JTAG reset event (EXTEST, HIGHZ, CLAMP instructions) and the software should not trigger a functional reset event via MC_ME or MDM_AP while the selftest is ongoing. This is to avoid any abort scenario due to any functional reset source as Self-Test is initiated by SW.

There is no issue with Reset pin as Reset pin is a source of destructive reset and will reset STCU2 and Self-Test related logic etc.

51.1.17 End of Self-Test

The Self-Test completion is a source of functional reset to the device, by which the device executes a functional reset sequence. The logic which is LBISTed additionally undergoes a complete POR sequence to ensure that the random content shifted during BIST sequence get cleared.

At the end of MBIST, the software should execute below steps:

1. Software reads the MBIST end flag to identify that the MBIST sequence had ended properly without any interruption or abort, by reading STCU2.MBESWn.
2. Software reads the MBIST status to identify whether the MBIST was successful or not, by reading STCU2.MBSSWn.

NOTE

The MBIST status should always be read in accordance with MBIST end flag. The STCU2.MBSSWn status is irrelevant if STCU2.MBESWn indicates that MBIST execution is incomplete.

At the end of LBIST, the software should execute below steps:

1. Software reads the LBIST end flag to identify that the LBIST sequence had ended properly without any interruption or abort, by reading STCU2.LBESWn.
2. Software reads the LBIST status to identify whether the LBIST was successful or not, by reading STCU2.LBSSWn.

NOTE

The LBIST status should always be read in accordance with LBIST end flag. The STCU2.LBSSWn status is irrelevant if STCU2.LBESWn indicates that LBIST execution is incomplete.

51.2 Introduction

STCU2 is a comprehensive programmable hardware module that:

- Manages the execution of BISTs
- Indicates whether each BIST passed
- Manages the chip's LBIST and the MBIST blocks

51.3 Main features

STCU2 includes:

- System interface for reading from and writing to the STCU2 registers during runtime (online) via the CPU
- Programmable scheduler for BIST execution
- Control over LBIST concurrent or sequential execution
- Control over MBIST concurrent or sequential execution
- Programmable LBIST delayed concurrent start
- Programmable internal clock prescaler to reduce internal and BIST clocks
- PLL lock signal monitoring during BIST sequences
- Programmable BIST-sequence-execution watchdog timer that specifies the maximum time allowed for the execution of a BIST sequence
- Fixed register-write-access watchdog timer that specifies the maximum amount of time that STCU2 allows you to write to its registers after you unlock them
- Fields that indicate the execution status of each BIST, see Types of BIST sequences and Types of BIST partitions
- Fields that indicate the status of each type of online STCU2 internal error condition

- Programmable fault mapping of each BIST for controlling the type of fault that STCU2 reports (recoverable or unrecoverable) when the BIST fails to execute
- Programmable fault mapping of each STCU2 internal error condition for controlling the type of fault that STCU2 reports (recoverable or unrecoverable) when the condition occurs
- Signals to report recoverable and unrecoverable faults to the FCCU module
- Redundant recoverable and unrecoverable fault-generation logic to improve reliability
- FCCU recoverable and unrecoverable fault-injection mechanism
- Global register write-protection mechanism that requires two security key codes
- Global automatic power saving after a BIST sequence is completed when watchdog timer time-out is detected
- Watchdog automatic clock wake-up mechanism when software unlocks the STCU2 registers for write access
- Watchdog automatic power saving when the fixed register-write-access watchdog timer times out

51.4 Block diagram

This diagram shows the parts of STCU2:

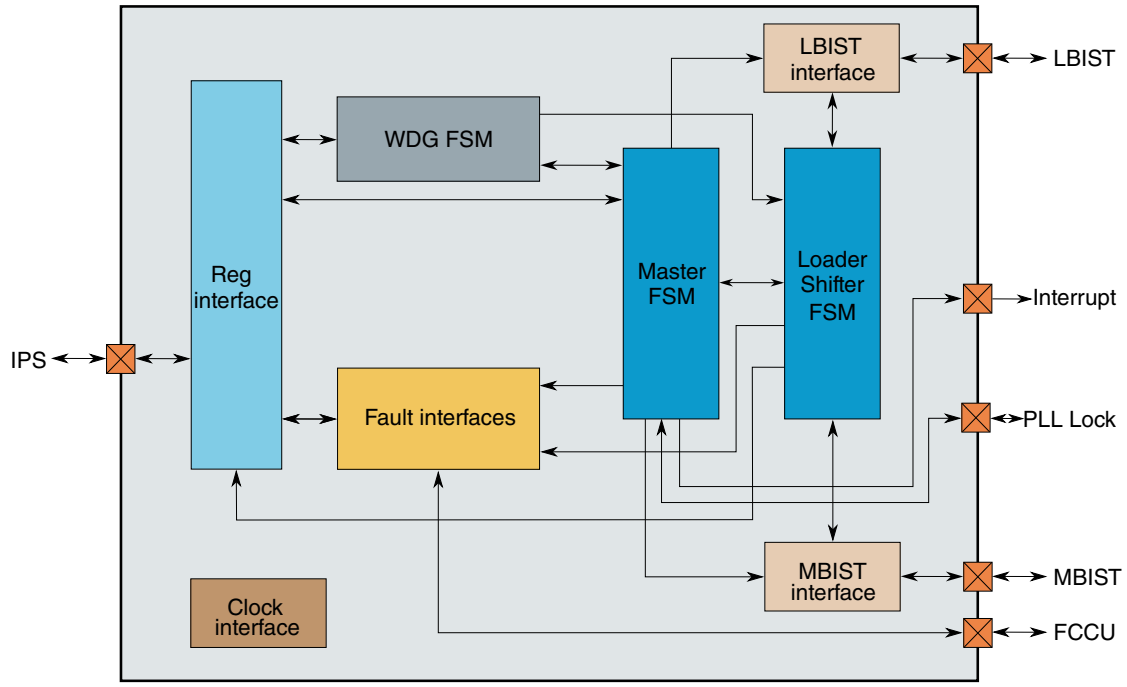


Figure 192. STCU2 block diagram

This table describes the parts of STCU2:

Table 244. STCU2 parts

Part	Function
Reg interface	Provides access to: • Registers

Table continues on the next page...

Table 244. STCU2 parts (continued)

Part	Function
	• The security key logic • The IPS interface
Fault interface	Performs the following tasks: • Collects the fault conditions caused by STCU2 internal error conditions and each BIST executed in a sequence • Sets the global recoverable or unrecoverable status flag, depending on the fault mapping for a given BIST • Manages the recoverable and unrecoverable fault lines to and from the FCCU module • Manages the set/clear injection mechanism provided by the FCCU module To improve the intrinsic reliability of this critical logic, the generation logic is duplicated.
Clock interface	Manages the internal and the BIST clock prescaler, the internal clock-gating power saving, and the wake-up clock feature
WDG FSM	Provides the following: • A programmable BIST-sequence-execution watchdog timer that specifies the maximum time allowed for the execution of a BIST sequence • A fixed register-write-access watchdog timer that specifies the maximum amount of time that STCU2 allows you to write to its registers after you unlock them
Master FSM	Coordinates and schedules all of the operations performed during a BIST sequence
Loader Shifter FSM	Programs the BIST registers and reads back the data to be checked at the end of each test operation
LBIST interface	Provides the interface between the chip's LBIST engines and the STCU2 controllers
MBIST interface	Provides the interface between the MBIST controller (MTR) and the STCU2 controllers

51.5 Peripheral bus interface

The peripheral bus interface is a slave bus used for configuration purposes via CPU. The module supports the following bus read operations:

- Word (32 bits) data read operations to any registers
- Any other operation is not supported.

Word, low and high half-words, byte operations are supported in write mode only:

- Write access is allowed by software only when CFG[WRP] is cleared. The default value of CFG[WRP] is cleared.
- Exceptions are the following read/write fields: CFG[WRP], ERR_STAT[UFSF] and ERR_STAT[RFSF].
- In all the other conditions, the write operations are not supported and all the STCU2 registers can be only read.

- STCU2 Configuration (CFG) should only be accessed in 32 bit (partial access not allowed).

The STCU2 module generates a transfer error in the following cases:

- Any read access to the registers after writing Key1 and before writing Key2 (in online) and before register write-access watchdog timer expires.
- Any write/read access to the register addresses not mapped on the peripheral but included in the address space of the peripheral
- Any write/read operation different from byte/halfword/word (free byte enables or other operations) on each register
- Any write operation on Double Security Key register applying a wrong sequence of keys (the two write operations cannot be interleaved with other access to STCU2 registers)
- Any write operation performed on a register when the Double Security keys have not been applied
- Any write operation performed on registers when CFG[WRP] is set and the access is performed through software (internal peripheral interface). The exceptions are SKC register (for valid keys value and sequence) and INT_FLG register(interrupt flag) in which STCU2 does not assert transfer error.
- Any write operation performed on Read Only registers

The registers of the STCU2 module are accessible (read/write) in each access mode: user, supervisor, or test.

In case there are write operations on bits marked as reserved, the transfer error is not generated.

NOTE

See the chip-specific STCU2 information for the wait time necessary to write to the online registers.

51.6 BISTs and BIST partitions

51.6.1 Definition: BIST

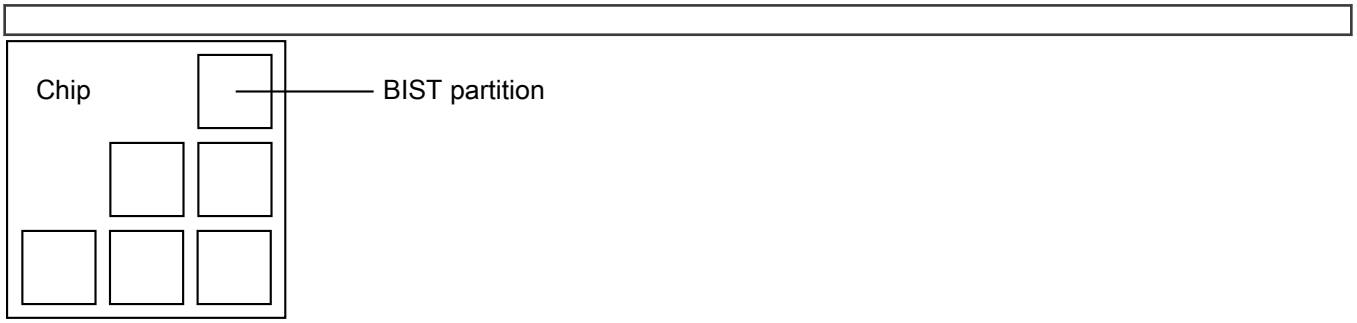
A BIST is a test that the chip can execute to verify the functional integrity of a part of itself. The chip uses STCU2 and other on-chip hardware to execute a BIST.

51.6.2 Definition: BIST partition

A BIST partition is a part of a chip for which a BIST has been defined. The hardware that is included in a given BIST partition is chip-specific. For a list of the hardware included in each of the BIST partitions on this chip, see the chip-specific STCU2 information.

51.6.3 Example: BIST partitions on a chip

This is an example of a chip with six BIST partitions:



51.6.4 Types of BIST partitions

The STCU2 module on this chip supports the following type or types of BIST partitions:

Table 245. Types of BIST partitions

BIST partition type	Description
MBIST partition	An SRAM or ROM block
LBIST partition	One or more digital modules

51.7 BIST sequences

51.7.1 Definition: BIST sequence

A BIST sequence is a programmable series of one or more phases, each of which executes one or more individual BISTs.

51.7.2 STCU2 executes MBISTs before LBISTs

If a BIST sequence includes both MBISTs and LBISTs; STCU2 executes the MBISTs first.

51.7.3 Example: Single-phase BIST sequence

This is an example of a single-phase BIST sequence in which STCU2 executes only one BIST:

Table 246. Example: Single-phase BIST sequence

Phase	BIST executed
0	MBIST 16

51.7.4 Example: Multi-phase BIST sequence

This is an example of a multiphase BIST sequence in which STCU2 executes more than one BIST in parallel in some phases:

Table 247. Example: Multi-phase BIST sequence

Phase	BISTs executed
0	MBISTs 7, 38
1	MBIST 23
2	MBISTs 6, 16, 29, 23
3	LBIST 12
4	LBISTs 13, 8, 11
5	LBISTs 19, 6

51.7.5 Types of BIST sequences

The STCU2 module on this chip supports these types of BIST sequences:

Table 248. Example: Types of BIST sequences

BIST sequence type	Description
Online BIST sequence	A BIST sequence that STCU2 executes during runtime at the request of software. Application software configures and initiates execution of the BIST sequence by loading values into STCU2 registers.

51.7.6 Supported BIST sequences

NXP supports only a specific set of BIST sequences for this chip. For the list of supported BIST sequences, see the chip-specific STCU2 information.

51.8 Functional description

51.8.1 FSM description

The module has three state machines that work together: Master State Machine, Loader/Shifter State Machine, and the Watchdog State Machine. Basically the Master State Machine is the core unit of the STCU2 module. It coordinates all the self-test operations and the other state machines. The Loader/Shifter State Machine is used to program the MBIST and the LBIST parameters and to retrieve the related results depending on the parameters stored into the STCU2 registers and under the control of the Master State Machine. The Watchdog State Machine evaluates all the schedule time for MBIST and LBIST and the time-out in case of wrong STCU2 programming.

51.8.2 BIST scheduling

STCU2 is designed to program the parallel/serial execution of the MBIST or LBIST depending on the power, timing, and coverage constraints.

The mechanism used to provide this flexibility is a linked list of BIST descriptors where the starting pointer is defined in CFG[PTR]. The first LBIST is mapped on 0, the second on 1, and so on. A BIST descriptor identifies a LBIST or MBIST via an index that is associated with its control register (STCU2 LBIST Control (LB_CTRL0) or STCU2 MBIST Control (MB_CTRL0 - MB_CTRL11)). The first MBIST is mapped on (00000080h + 0), the second on (00000080h + 1), and so on. The additional pointers of the linked list are in LB_CTRL*n*[PTR] for LBIST *n* (where *n* is the selected LBIST) and MB_CTRL*m*[PTR] for MBIST *m* (where *m* is the selected MBIST) and must be populated depending on the selected run sequence. The additional fields LB_CTRL[CSM] and MB_CTRL[CSM] provide the flexibility to run concurrently or sequentially the chosen set of the LBIST or MBIST or to close the linked list by setting the NIL pointer.

51.8.3 FCCU interface

The FCCU interface is the hardware flag mechanism towards the system, to indicate the occurrence of an Unrecoverable fault and/or a Recoverable fault failure during the Self Test sequence.

To diagnose physical defects on the two fault signals, a fault injection mechanism is also provided. In this case, the FCCU interface allows the user application to check the integrity of the UF and RF connection lines between the STCU2 and the FCCU. Refer to the description of FCCU fault injection mechanism to understand how the UF/RF set/clear mechanism works.

51.8.4 Watchdogs

The STCU2 implements different watchdogs to ensure that operations are finished in time.

51.8.4.1 BIST watchdog timer

The LBIST and MBIST execution time has to be configured as described in STCU2 Watchdog Granularity (WDG), to account for the overall execution time of the self test sequence. In case the selected LBISTs or MBISTs are not yet completed during assigned

time, the current LBISTs or MBISTs execution is interrupted and a failure is flagged into ERR_STAT[WDTOSW] and MBESWx or LBESW.

In case of multiple sequential run in the same online session and the time-out happens in the middle of a sequential run, the next sequential run will be skipped and the execution ends with the current updated status of the registers reported above.

51.8.4.2 Register write-access watchdog timer

As explained in the STCU2 SK Code (SKC):

- A key mechanism protects STCU2 registers during the self-test configuration phase by preventing any unwanted access.
- A hardware watchdog timer locks register-write access after a number of STCU2 clock cycles. To refresh the hardware watchdog timer before it times out, write Key1 and Key2 sequences.

AUTOLOCK_VALUE is the number of STCU2 clock cycles after which the hardware watchdog timer locks register-write access. See the chip-specific STCU2 information for the value of AUTOLOCK_VALUE.

The hardware watchdog timer is particularly useful in case CFG[WRP] is 0 during the software self-test configuration. In this case, the software application might enable write access to the STCU2 registers.

51.9 Use cases and limitations

For details on use cases and limitations, contact NXP sales representative.

51.10 STCU2 register descriptions

The STCU2 registers are listed in this section.

NOTE

Always write the reset value for the Reserved fields.

During online self-test, after the STCU2 registers are configured, they must not be overridden via internal peripheral system until the self-test is complete.

51.10.1 STCU2 memory map

STCU base address: 403A_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	STCU2 Run Software (RUNSW)	32	RW	0000_0000h
8h	STCU2 SK Code (SKC)	32	W	0000_0000h
Ch	STCU2 Configuration (CFG)	32	RW	0000_0000h
14h	STCU2 Watchdog Granularity (WDG)	32	RW	0000_FFFFh
24h	STCU2 Error (ERR_STAT)	32	RW	0000_0000h
28h	STCU2 Error FM (ERR_FM)	32	RW	0000_0000h
4Ch	STCU2 Online LBIST Status (LBSSW0)	32	R	0000_0000h
5Ch	STCU2 Online LBIST End Flag (LBESW0)	32	R	0000_0000h
7Ch	STCU2 Online LBIST Unrecoverable FM (LBUFM0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10Ch	STCU2 Online MBIST Status (MBSSW0)	32	R	0000_0000h
14Ch	STCU2 Online MBIST End Flag (MBESW0)	32	R	0000_0000h
18Ch	STCU2 MBIST Unrecoverable FM (MBUFM0)	32	RW	0000_0000h
200h	STCU2 LBIST Control (LB_CTRL0)	32	RW	0000_0000h
204h	STCU2 LBIST PC Stop (LB_PCS0)	32	RW	0000_0000h
220h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW0)	32	RW	FFFF_FFFFh
224h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW0)	32	RW	FFFF_FFFFh
228h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW0)	32	R	0000_0000h
22Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW0)	32	R	0000_0000h
2200h	STCU2 Algorithm Select (ALGOSEL)	32	RW	0000_0000h
220Ch	STCU2 MBIST Stagger (STGGR)	32	RW	0000_0000h
2210h	STCU2 BIST Start (BSTART)	32	RW	0000_0000h
2214h - 2240h	STCU2 MBIST Control (MB_CTRL0 - MB_CTRL11)	32	RW	0000_0000h

51.10.2 STCU2 Run Software (RUNSW)

Offset

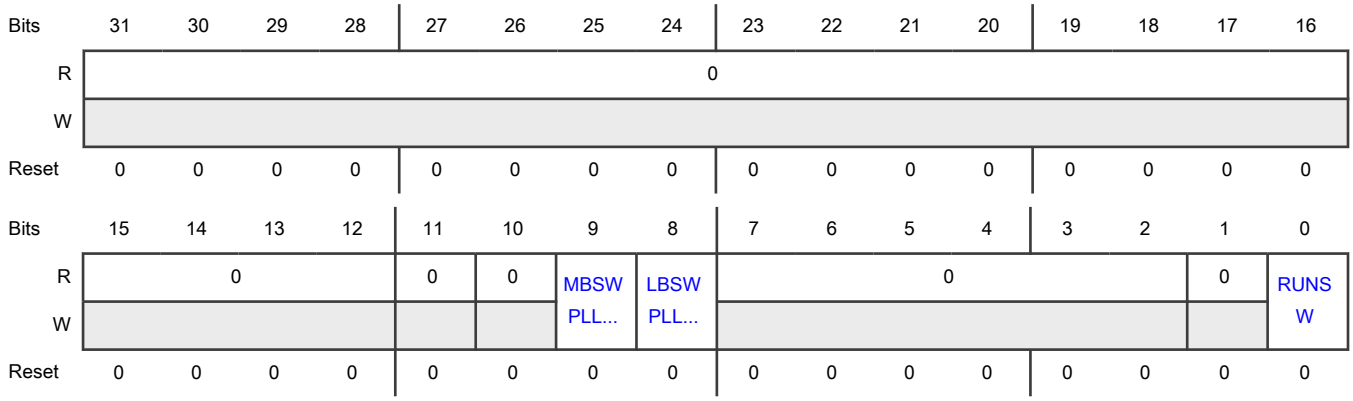
Register	Offset
RUNSW	4h

Function

The RUNSW register defines the RUN bit to start the online self-testing procedure.

The R/W fields in this register are readable at any time. However, you can write to these fields only when CFG[WRP] = 0.

Diagram



Fields

Field	Function
31-12 —	Reserved
11 —	Reserved
10 —	Reserved
9 MBSWPLLEN	Online MBIST with PLL Enabled 0b - Online MBIST is executed without using the on-chip PLL. 1b - Online MBIST is executed using the PLL configuration provided by software. STCU2 does not take the PLL control but monitors the PLL lock signal to check if PLL is working correctly.
8 LBSWPLLEN	Online LBIST with PLL Enabled 0b - Online LBIST is executed without using the on-chip PLL. 1b - Online LBIST is executed using the PLL configuration provided by the software. STCU2 does not take the PLL control but monitors the PLL lock signal to check if PLL is working correctly.
7-2 —	Reserved
1 —	Reserved
0 RUNSW	The RUNSW bit is automatically cleared by STCU2 when the online self-testing procedure is complete. 0b - Idle 1b - Online self-testing procedure is running

51.10.3 STCU2 SK Code (SKC)

Offset

Register	Offset
SKC	8h

Function

The SKC register implements the security key code mechanism needed to access in write mode to the other STCU2 registers.

To unlock STCU2 access after the STCU2 asynchronous reset and at the end of the STCU2 run, the software (IPS bus) need to apply the following sequence:

- write the key1 into the SKC register
- write the key2 into the SKC register

Depending on the online test step, the two keys are different. Byte write operation is not allowed because the full key has to be recognized as one unit.

In case of invalid access or sequence (Key1/2 have to be applied consecutively), a transfer error on the IPS is asserted. The STCU2 write access is locked and to unlock the access, the sequence has to be applied again.

In case the STCU2 register access lasts more cycles than the one defined in the hard-coded WDG timeout, the STCU2 write access is locked and the WDG and register interface clocks are switched off. Also, in this case, to enable the write access to the STCU2 again and the WDG and register interface clocks, it is required to apply the sequence again. This hard-coded WDG counter starts decrementing its value right after POR is deasserted.

In case it is required to extend the STCU2 register access cycles before the hard-coded WDG timeout expires, Key1 and Key2 sequences need to be applied. The effect of this write operation is to re-initialize the WDG timeout counter. The STCU2 write access is locked and to unlock the access, the sequence has to be applied again.

The SKC register is not readable. The value 00000000h is always returned in case of read operation.

Diagram

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

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

Fields

Field	Function
31-0	STCU2 SK Code

Table continues on the next page...

Field	Function
SKC	STCU2 security key code for online test = 753F924Eh: Key1 to unlock the write access the STCU2 = 8AC06DB1h: Key2 to unlock the write access the STCU2

51.10.4 STCU2 Configuration (CFG)

Offset

Register	Offset
CFG	Ch

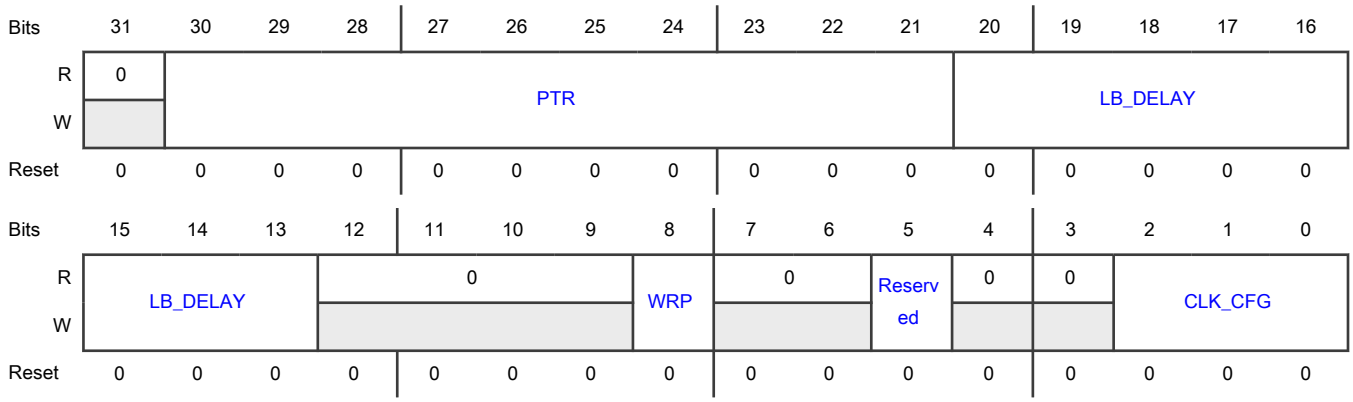
Function

The CFG register includes the global configuration of the STCU2 and can be updated in the online test steps.

The access to this register is described in the following figure. It further depends on the state of the WRP bit as follows:

- When WRP = 0: The register can be written during the online self-test case without restrictions.
- When WRP = 1:
  - The only bit that can be written without any restriction is WRP.
  - In case there are software operations that write all the register's bits, a transfer error is raised only if the value of the selected byte written differs from the current status of the register. This functionality has been implemented to prevent potential compilation behavior that might invalidate this single bit clean capability.

Diagram



Fields

Field	Function
31 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-21 PTR	First LBIST or MBIST pointer PTR defines the logical pointer to the first LBIST or MBIST to be scheduled when the self-testing procedure is enabled. PTR is the entry pointer to a linked list of BIST descriptors. If PTR = 0 then the LBIST0 is initially scheduled. See BIST scheduling for details. **NOTE** You can write to this field only when WRP = 0. 0h to (01h - 1): pointer to LBIST 00000080h to (00000080h + 00Ch - 1): pointer to MBIST 3FFh: pointer to NIL. No BIST execution. others: invalid pointer => an error is set into the STCU2 Error (ERR_STAT).
20-13 LB_DELAY	Delay LBIST run LB_DELAY defines the delay between the LBIST starts when more than a single LBIST is selected to be executed concurrently with the purpose of smoothing the power consumption transient. The allowed delay time are these: 00h: No delay 01h: 1 x 16 STCU2 CORE_CLK cycles 02h: 2 x 16 STCU2 CORE_CLK cycles 03h: 3 x 16 STCU2 CORE_CLK cycles ... FDh: 253 x 16 STCU2 CORE_CLK cycles FEh: 254 x 16 STCU2 CORE_CLK cycles FFh: 255 x 16 STCU2 CORE_CLK cycles **NOTE** You can write to this field only when WRP = 0.
12-9 —	Reserved
8 WRP	Write Protection 0: Specific STCU2 registers can be written through IPS bus interface 1: STCU2 registers cannot be written through IPS, preventing any user application write operation
7-6 —	Reserved
5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 —	Reserved
3 —	Reserved
2-0 CLK_CFG	Logic, Memory BIST, and STCU2 CORE_CLK configuration CLK_CFG defines the ratio between the sys_clk and the internal clock used to program both the LBIST and the MBIST and the STCU2 CORE_CLK. The punch-out mechanism is used to generate the derived clocks. The following configurations are allowed: **NOTE** You can write to this field only when WRP = 0. 000b - sys_clk/1 001b - sys_clk/2 010b - sys_clk/3 011b - sys_clk/4 100b - sys_clk/5 101b - sys_clk/6 110b - sys_clk/7 111b - sys_clk/8

51.10.5 STCU2 Watchdog Granularity (WDG)

Offset

Register	Offset
WDG	14h

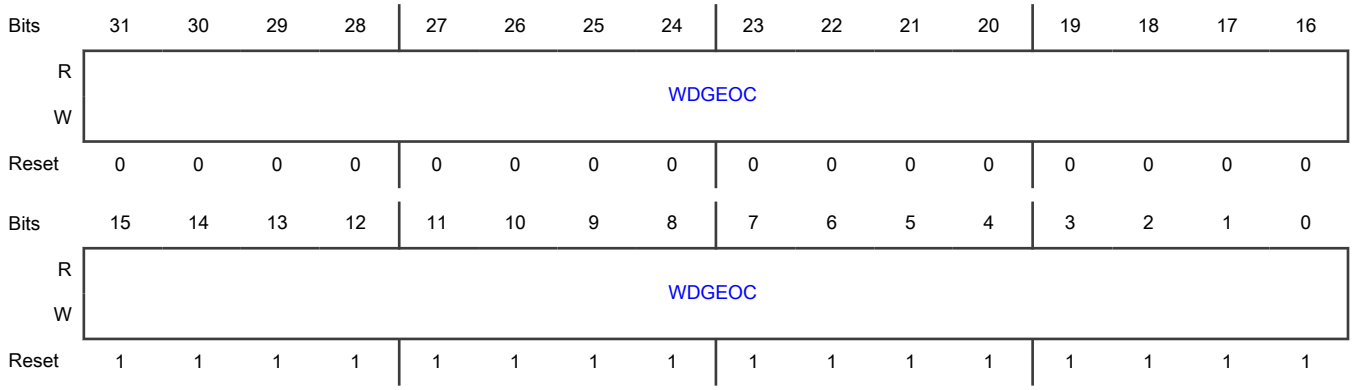
Function

The WDG register defines the time budget of LBIST and MBIST execution providing a protection mechanism in case of dead-lock or endless conditions during the self-test procedure.

In case online self-test sequence is run, it defines the timeout of the execution run.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram



Fields

Field	Function
31-0 WDGEOC	Watchdog End of Count Timer This value has to be set to define the time budget related to the online self-test execution and check that everything is correctly working within this slot of time. The delay time slots that are allowed are as follows: 0000 0000h: 1 x 16 STCU2 CORE_CLK cycles 0000 0001h: 2 x 16 STCU2 CORE_CLK cycles 0000 0002h: 3 x 16 STCU2 CORE_CLK cycles ... FFFF FFFDh: 4294967294 x 16 STCU2 CORE_CLK cycles FFFF FFFEh: 4294967295 x 16 STCU2 CORE_CLK cycles FFFF FFFFh: 4294967296 x 16 STCU2 CORE_CLK cycles

51.10.6 STCU2 Error (ERR_STAT)

Offset

Register	Offset
ERR_STAT	24h

Function

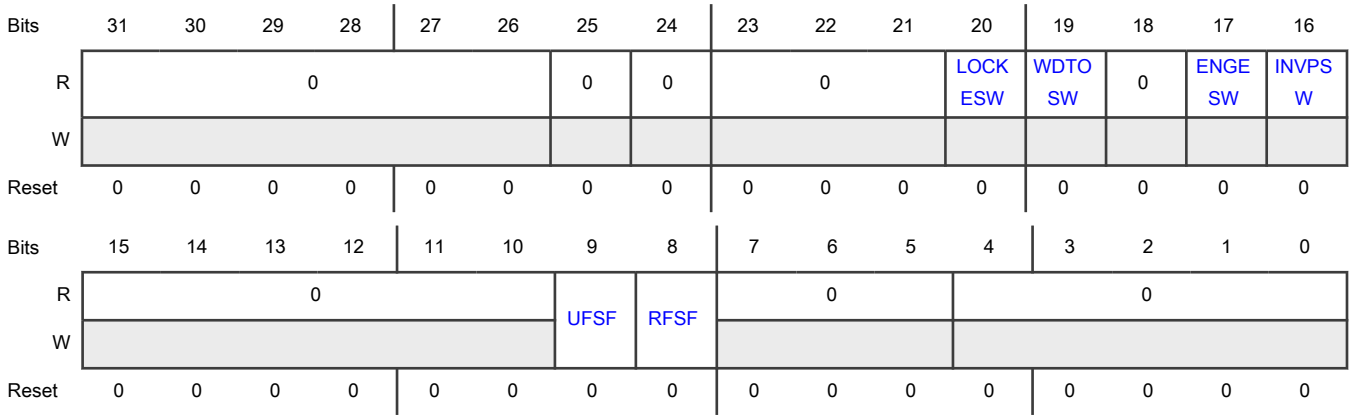
The ERR_STAT register includes the status flags related to the STCU2 internal error conditions occurred during the configuration or the online self-testing execution.

The UFSF and RFSF can be set/cleared using the FCCU dedicated channels.

The access to this register is described in the following figure and as follows:

- If you select the byte write capability to write only UFSF and RFSF, then there is no restriction in writing these bits.
- If your software performs the write operations on other bits besides UFSF/RFSF, then a transfer error is generated only if the value you are writing to the other bits differs from their value currently stored in the register. This functionality has been implemented to prevent potential compilation behavior that might invalidate the UFSF/RFSF single bit set/clean capability.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 —	Reserved
24 —	Reserved
23-21 —	Reserved
20 LOCKESW	Online LOCK error You can always read this field. The content of this field is initialized to its reset value when RUNSW[RUNSW] is set to 1. 0b - In case PLL is enabled, it is correctly locked during the self-test sequence 1b - When the PLL is enabled, this flag highlights that there has been an unexpected PLL unlock(loss-of-lock) event during the online self-test sequence execution. The online self-test run is stopped and the status of the currently running LBISTs or MBISTs is saved in the related registers. The LOCK signal is monitored during the LBIST run when RUNSW[LBSWPLLEN] is set and/or during the MBIST run when RUNSW[MBSWPLLEN] = 1.
19 WDTOSW	Online watchdog timeout You can always read this field. The content of this field is initialized to its reset value when RUNSW[RUNSW] is set to 1. 0b - LBIST and MBIST time slots completed within the assigned watchdog time. 1b - LBIST and MBIST time slots not completed within the assigned watchdog time or there are internal mismatches among End of Execution signals.

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 —	Reserved
17 ENGESW	Online engine error You can always read this field. The content of this field is initialized to its reset value when RUNSW[RUNSW] is set to 1. 0b - Valid engine execution 1b - Invalid engine execution. The error conditions that set this bit are FSM, protocol error, and so on.
16 INVPSW	Online invalid pointer You can always read this field. The content of this field is initialized to its reset value when RUNSW[RUNSW] is set to 1. 0b - Valid linked pointer list 1b - Invalid linked pointer list. The following conditions set this bit: Initial LBIST orMBIST pointer is out of range; LBIST is selected when MBIST is concurrently running or vice versa; Error in the LBIST/MBIST linking (execution generates an infinite loop).
15-10 —	Reserved
9 UFSF	Unrecoverable Faults Status Flag This flag reports the global status of the Unrecoverable Faults(UF). This field can be set or cleared using the FCCU dedicated channel, and can also be set or cleared by software. 0b - No errors that trigger the UF condition. 1b - There are errors that trigger the UF condition.
8 RFSF	Recoverable Faults Status Flag This flag reports the global status of the Recoverable Fault (RF). This field can be set or cleared using the FCCU dedicated channel, and can also be set or cleared by software. 0b - No errors that trigger the Recoverable Faults condition 1b - There are errors that trigger the Recoverable Faults condition
7-5 —	Reserved
4-0 —	Reserved Write may occur on these fields but writes have no effect.

51.10.7 STCU2 Error FM (ERR_FM)

Offset

Register	Offset
ERR_FM	28h

Function

The ERR_FM register defines the fault mapping of the STCU2 faults described in the register ERR_STAT in terms of UF or RF. All sources of internal faults can be routed to UF and RF.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram

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

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

Fields

Field	Function
31-5 —	Reserved
4 LOCKEUFM	PLL LOCK Unrecoverable Fault Mapping 0b - Recoverable Fault Mapping 1b - Unrecoverable Fault Mapping
3 WDTOUFM	Watchdog Timeout Unrecoverable Fault Mapping 0b - Recoverable Fault Mapping 1b - Unrecoverable Fault Mapping
2 —	Reserved
1	Engine Error Unrecoverable Fault Mapping

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENGEUFM	0b - Recoverable Fault Mapping 1b - Unrecoverable Fault Mapping
0 INVPUFM	Invalid Pointer Unrecoverable Fault Mapping 0b - Recoverable Fault Mapping 1b - Unrecoverable Mapping

51.10.8 STCU2 Online LBIST Status (LBSSW0)

Offset

Register	Offset
LBSSW0	4Ch

Function

This register includes the results corresponding to the execution of the selected online LBIST.

The size of the register depends on the number of LBIST .

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															LBSSW0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0-0 LBSSWn	LBSSWn online status of the selected LBIST 0b - Failed LBIST execution 1b - Successful LBIST execution

51.10.9 STCU2 Online LBIST End Flag (LBESW0)

Offset

Register	Offset
LBESW0	5Ch

Function

This register includes the results corresponding to the execution of the selected online LBIST.

The size of the register depends on the number of LBIST .

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															LBESW0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0-0 LBESWn	LBESW LBESWx: online LBIST end status 0b - LBIST execution not yet completed 1b - LBIST execution finished

51.10.10 STCU2 Online LBIST Unrecoverable FM (LBUFM0)

Offset

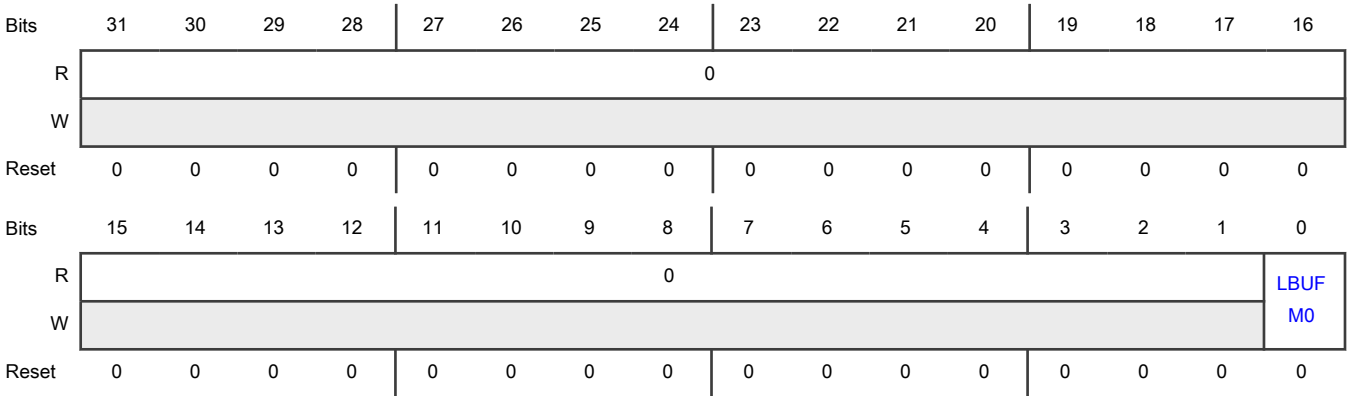
Register	Offset
LBUFM0	7Ch

Function

This register includes the results corresponding to the execution of the selected online LBIST.

The size of the register 0 depends on the number of LBIST .

Diagram



Fields

Field	Function
31-1 —	Reserved
0-0 LBUFMn	LBIST Unrecoverable Fault Mapping 0b - Recoverable Fault mapping 1b - Unrecoverable Fault mapping

51.10.11 STCU2 Online MBIST Status (MBSSW0)

Offset

Register	Offset
MBSSW0	10Ch

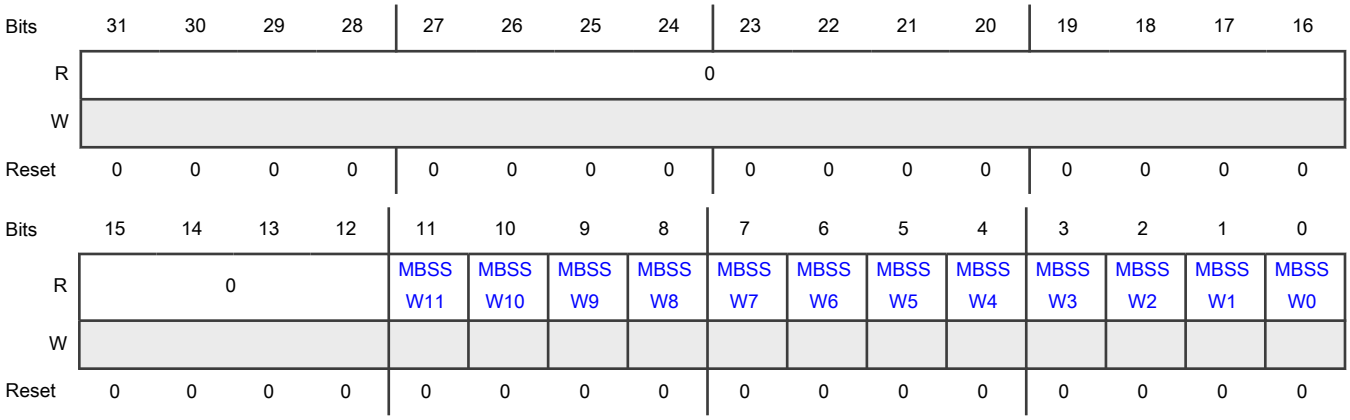
Function

This register includes the results corresponding to the execution of the selected online MBIST in the range 0:11.

The size of the register depends on the number of BISTed RAMs/ROMs.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 MBSSWn	MBSSW Online status of MBISTn (where n = 11:0). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution

51.10.12 STCU2 Online MBIST End Flag (MBESW0)

Offset

Register	Offset
MBESW0	14Ch

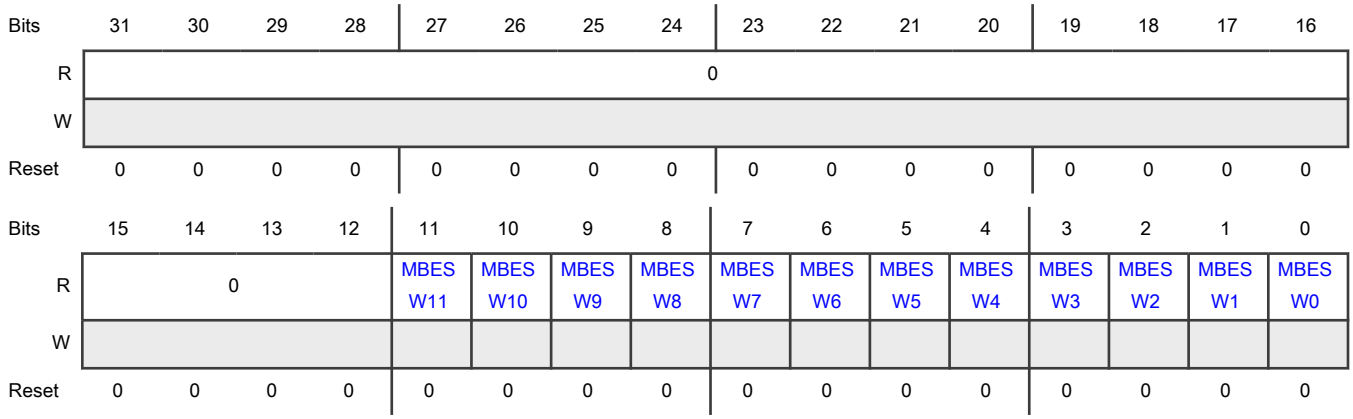
Function

This register includes the End Flag related to the execution of the selected online MBIST in the range 0:11.

The size of the register depends on the number of BISTed RAMs/ROMs.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 MBESWn	Online end status of MBISTn (where n = 11:0). 0b - MBIST execution still ongoing 1b - MBIST execution finished

51.10.13 STCU2 MBIST Unrecoverable FM (MBUFM0)

Offset

Register	Offset
MBUFM0	18Ch

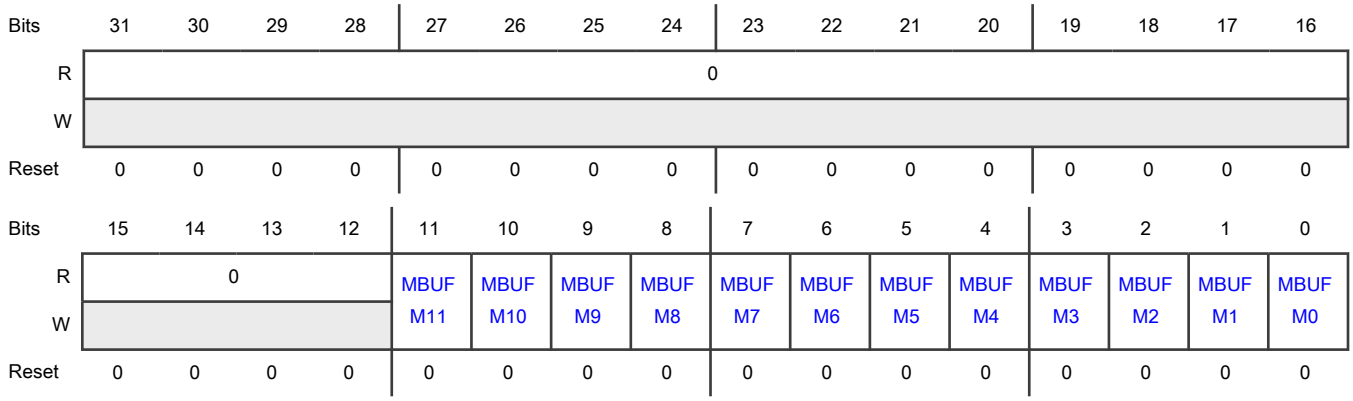
Function

This register defines the fault mapping, in terms of UF or RF, of the MBIST in the range 0:11

The size of the register depends on the number of BISTed RAMs/ROMs.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 MBUFMn	MBESW Online end status of MBISTn (where n = 11:0). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping

51.10.14 STCU2 LBIST Control (LB_CTRL0)

Offset

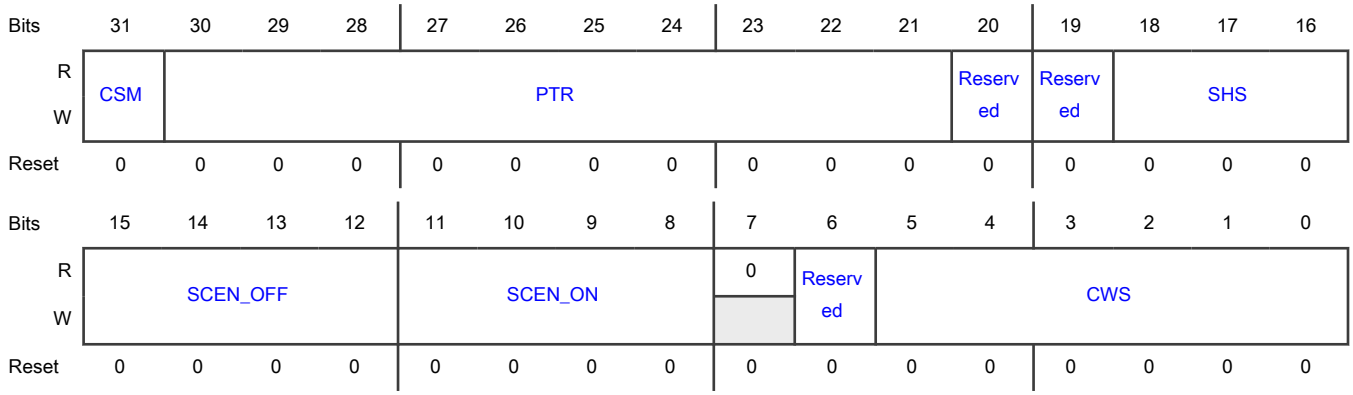
Register	Offset
LB_CTRL0	200h

Function

This register defines the control setting of each LBIST controller.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram



Fields

Field	Function
31 CSM	Concurrent/sequential mode The next LBIST is scheduled concurrently to the current one if the CSM bit is set to 1; otherwise, it is scheduled sequentially to the completion of the current LBIST execution. 0b - Sequential mode 1b - Concurrent mode
30-21 PTR	Next LBIST or MBIST pointer PTR defines the logical pointer to the next LBIST or MBIST to be scheduled. The next LBIST or MBIST is scheduled concurrently to the current one if the CSM bit is set to 1, otherwise it is scheduled sequentially to the completion of the current LBIST execution. In case of NIL pointer, the CSM bit has to be set Sequential (0) to define this is the end of the list. The self-testing procedure stops after the last BIST in the configuration chain is complete. See BIST scheduling for details. 0h to (01h - 1): pointer to NLBIST-1 00000080h to (00000080h + 00Ch - 1): pointer to MBIST 3FFh: pointer to NIL. No BIST execution. others: invalid pointer => an error is set into the STCU2 Error (ERR_STAT).
20 —	Reserved **NOTE** This reserved field is writable but do not write any value to it other than its reset value
19 —	Reserved **NOTE** This reserved field is writable but do not write any value to it other than its reset value
18-16 SHS	Shift speed SHS defines the shift speed

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - Shift at full rate (BIST clock). 001b - Shift at 1/2 rate (BIST clock). 010b - Shift at 1/3 rate (BIST clock). 011b - Shift at 1/4 rate (BIST clock). 100b - Shift at 1/5 rate (BIST clock). 101b - Shift at 1/6 rate (BIST clock). 110b - Shift at 1/7 rate (BIST clock). 111b - Shift at 1/8 rate (BIST clock).
15-12 SCEN_OFF	Scan enable OFF SCEN_OFF information is used to configure the lbist controller hardware to generate off_cycles, delay cycles during the scan enable off transition. **NOTE** SCEN_OFF must be programmed to a value >=1. 0000b - 0 delay cycles 0001b - 1 delay cycle 0010b - 2 delay cycles 0011b - 3 delay cycles 0100b - 4 delay cycles 0101b - 5 delay cycles 0110b - 6 delay cycles 0111b - 7 delay cycles 1000b - 8 delay cycles 1001b - 9 delay cycles 1010b - 10 delay cycles 1011b - 11 delay cycles 1100b - 12 delay cycles 1101b - 13 delay cycles 1110b - 14 delay cycles 1111b - 15 delay cycles
11-8 SCEN_ON	Scan enable ON SCEN_ON information is used to configure the lbist controller hardware to generate on_cycles, delay cycles during the scan enable on transition,

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE SCEN_ON delay register value must be programmed to a value >=1 0000b - 0 delay cycles 0001b - 1 delay cycle 0010b - 2 delay cycles 0011b - 3 delay cycles 0100b - 4 delay cycles 0101b - 5 delay cycles 0110b - 6 delay cycles 0111b - 7 delay cycles 1000b - 8 delay cycles 1001b - 9 delay cycles 1010b - 10 delay cycles 1011b - 11 delay cycles 1100b - 12 delay cycles 1101b - 13 delay cycles 1110b - 14 delay cycles 1111b - 15 delay cycles
7 —	Reserved
6 —	Reserved
5-0 CWS	Capture window size CWS defines the capture window size. 00_0000b - Illegal 00_0001b - Controller waits 1 shift cycle for capture to finish. 00_0010b - Controller waits 2 shift cycles for capture to finish. 00_0011b - Controller waits 3 shift cycles for capture to finish. 00_0100b - Controller waits 4 shift cycles for capture to finish. 00_0101b - Controller waits 5 shift cycles for capture to finish. 00_0110b - Controller waits 6 shift cycles for capture to finish. 00_0111b - Controller waits 7 shift cycles for capture to finish.

51.10.15 STCU2 LBIST PC Stop (LB_PCS0)

Offset

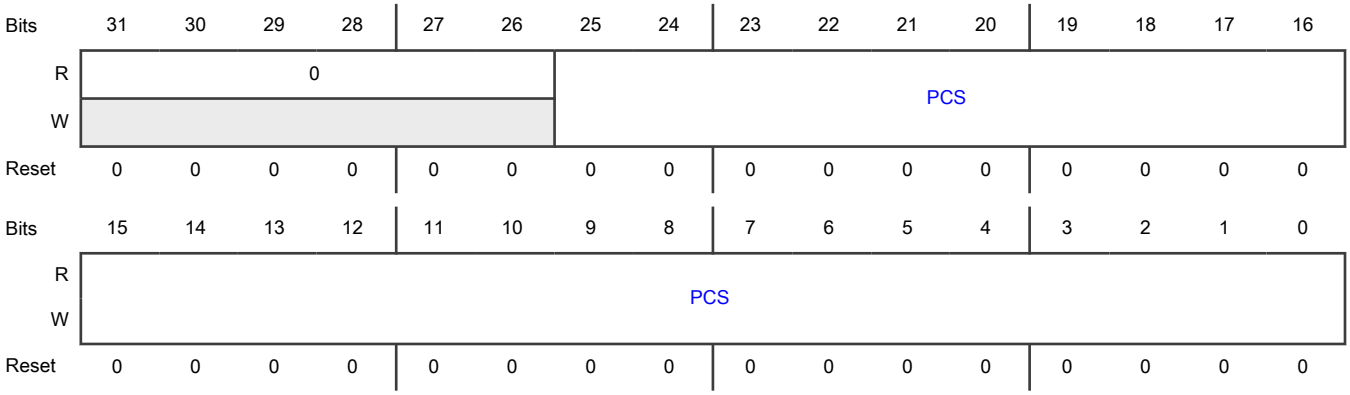
Register	Offset
LB_PCS0	204h

Function

This register defines the pattern counter stop of each LBIST controller.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-0 PCS	PCS Pattern counter stop PCS defines the pattern counter stop value.

51.10.16 STCU2 Online LBIST MISR Expected Low (LB_MISRELSW0)

Offset

Register	Offset
LB_MISRELSW0	220h

Function

This register defines bits 32 of the expected MISR of the online LBIST controller.

The size of the register depends on the number of MISR bits of the related LBIST. .

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram

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

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

Fields

Field	Function
31-0 MISRESWx	Online MISR expected low bits This field defines 32 bits of the expected MISR.

51.10.17 STCU2 Online LBIST MISR Expected High (LB_MISREHSW0)

Offset

Register	Offset
LB_MISREHSW0	224h

Function

The size of the register depends on the number of MISR bits of the related LBIST. .

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram

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

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

Fields

Field	Function
31-0 MISRESWx	Online MISR Expected High Bits This field defines the 32 bits of the expected MISR.

51.10.18 STCU2 Online LBIST MISR Read Low (LB_MISRRLSW0)

Offset

Register	Offset
LB_MISRRLSW0	228h

Function

This register reports 32 bits of the MISR obtained at the end of the online LBIST controller execution.

The size of the register depends on the number of MISR bits of the related LBIST. .

The content of this register is initialized to its reset value reset value when RUNSW[RUNSW] is set to 1.

Diagram

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

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

Fields

Field	Function
31-0 MISRRSWx	MISRRSWx Online MISR Read Low Bin This field is equivalent to 32 bits of the MISR obtained at the end of the online LBIST execution.

51.10.19 STCU2 Online LBIST MISR Read High (LB_MISRRHSW0)

Offset

Register	Offset
LB_MISRRHSW0	22Ch

Function

The size of the register depends on the number of MISR bits of the related LBIST. .

The content of this register is initialized to its reset value reset value when RUNSW[RUNSW] is set to 1.

Diagram

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

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

Fields

Field	Function
31-0 MISRRSWx	MISRRSWx Online MISR Read High Bits This field is equivalent to 32 bits of the MISR obtained at the end of the online LBIST execution.

51.10.20 STCU2 Algorithm Select (ALGOSEL)

Offset

Register	Offset
ALGOSEL	2200h

Function

This is a 32-bit register intended to be programmed by the user to select algorithms to be run on BIST. See the chip-specific STCU2 information for details of this register.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL9	ALGO SEL8	ALGO SEL7	ALGO SEL6	ALGO SEL5	ALGO SEL4	ALGO SEL3	ALGO SEL2	ALGO SEL1	ALGO SEL0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ALGOSELn	Algorithm Select

51.10.21 STCU2 MBIST Stagger (STGGR)

Offset

Register	Offset
STGGR	220Ch

Function

This register allows one to program number of clock cycles between execution of one BIST and the next one.

Diagram

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

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

Fields

Field	Function
31-0 STAG	STAG Number of STCU2 CORE_CLK cycles between execution of one BIST and the next one.

51.10.22 STCU2 BIST Start (BSTART)

Offset

Register	Offset
BSTART	2210h

Function

This is a 32-bit register intended to be programmed by the user to run BISTs with different configuration. See the chip-specific STCU2 information for details of this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BSTART31	BSTART30	BSTART29	BSTART28	BSTART27	BSTART26	BSTART25	BSTART24	BSTART23	BSTART22	BSTART21	BSTART20	BSTART19	BSTART18	BSTART17	BSTART16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BSTART15	BSTART14	BSTART13	BSTART12	BSTART11	BSTART10	BSTART9	BSTART8	BSTART7	BSTART6	BSTART5	BSTART4	BSTART3	BSTART2	BSTART1	BSTART0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BSTARTn	BIST Start

51.10.23 STCU2 MBIST Control (MB_CTRL0 - MB_CTRL11)

Offset

For a = 0 to 11:

Register	Offset
MB_CTRLa	2214h + (a × 4h)

Function

The MB_CTRL register defines the control setting of MBIST controller.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram

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

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

Fields

Field	Function
31 CSM	CSM Concurrent/sequential mode 0b - Sequential mode 1b - Concurrent mode
30-21	PTR

Table continues on the next page...

Table continued from the previous page...

Field	Function
PTR	Next LBIST or MBIST pointer PTR defines the logical pointer to the next LBIST or MBIST to be scheduled. The next LBIST orMBIST is scheduled concurrently to the current one if the CSM bit is set to 1; otherwise it is scheduled sequentially to the completion of the current MBIST execution. In case of NIL pointer the CSM bit must be set Sequential (0) to define this is the end of the list. The self-testing procedure stops after the last BIST in the configuration chain is complete. See BIST scheduling for details. **NOTE** If the pointer is invalid and MBIST is scheduled to run concurrently than this invalid scenario is handled by watchdog timeout feature and corresponding watchdog timeout status bit will be updated in ERR_STAT[WDTOSW]. In this particular case ERR_STAT[INVPSW] will not be set. 0h to (01h - 1): pointer to LBIST 00000080h to (00000080h + 00Ch - 1): pointer to MBIST 3FFh: pointer to NIL. No next BIST execution. others: invalid pointer => an error is set into the STCU2 Error (ERR_STAT).
20 BSEL	BSEL BIST Select 0b - Selected BIST is not selected for execution. 1b - Selected BIST is selected for execution.
19-0 —	Reserved

51.11 Glossary

BIST	Built-in self-test
BIST Clock	BIST controller clock corresponding to the specific BIST
CORE_CLK	Clock specified by the CFG register
FSM	Finite state machine
IPS	Internal peripheral system
LBIST	Logic BIST
MBIST	Memory BIST
NLBIST	Number of logic built-in self-test controller
NMCUT	Number of memory checked using memory built-in self-test controller
RF	Recoverable faults
UF	Unrecoverable faults
WDG FSM	Watchdog finite state machine

Chapter 52
Register Protection (REG_PROT)

52.1 Chip-specific REG_PROT information

52.1.1 REG_PROT configuration

The MCU safety relevant configuration registers are protected against unauthorized HW/S changes during read/write/clear access by implementing them with register protection module. See the REG_PROT details file attached to this document.

52.1.1.1 CMU_BRIDGE register protection

The CMUs in the device reside on a common peripheral slot as CMU_BRIDGE in the system memory map. Figure 193 indicates the locations of the CMUs alongwith the memory map.

Based on the CMU location in system peripheral memory map, the below locations in the CMU 16KB space are reserved and any access on these locations results in a bus transfer error.

402B_C018 – 402B_C01F

402B_C030 – 402B_C03F

402B_C050 – 402B_C05F

402B_C078 – 402B_C07F

402B_C098 – 402B_C09F

402B_C0B8 – 402B_CFFF

Since the register protection is available in the device for CMU_BRIDGE, the 4KB offset locations (refers as the Area 2 in register protection) are also reserved and accesses over these regions also result in bus transfer error.

402B_D018 – 402B_D01F

402B_D030 – 402B_D03F

402B_D050 – 402B_D05F

402B_D078 – 402B_D07F

402B_D098 – 402B_D09F

402B_D0B8 – 402B_DFFF

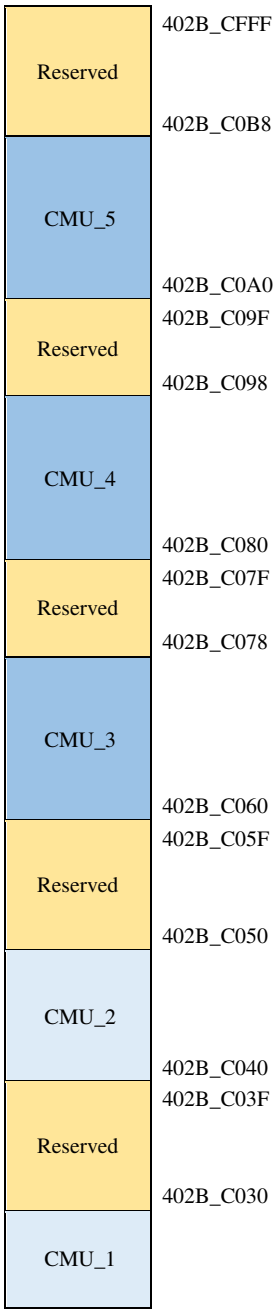


Figure 193. Location of CMUs

52.2 Overview

REG_PROT offers a mechanism to protect defined memory-mapped address locations, in a module under protection, from being written.

REG_PROT is a protection module that is located between the module under protection and the peripheral interface. The address locations to be protected exist in the module under protection and the address locations that can be protected are module specific. Writes to these protected address locations are locked using a Soft Lock Bit Register (SLBR*n*). Any access to address locations in the module under protection is restricted if their corresponding Soft lock fields in a Soft Lock Bit Register (SLBR*n*) are 1.

The configured soft-lock fields can also be write-restricted by using GCR[HLB]. When GCR[HLB]=1, you cannot write to the soft-lock fields. The address locations in the module under protection can be restricted to Supervisor mode access using GCR[UAA].

52.2.1 Block Diagram

The following figure shows the block diagram of this module.

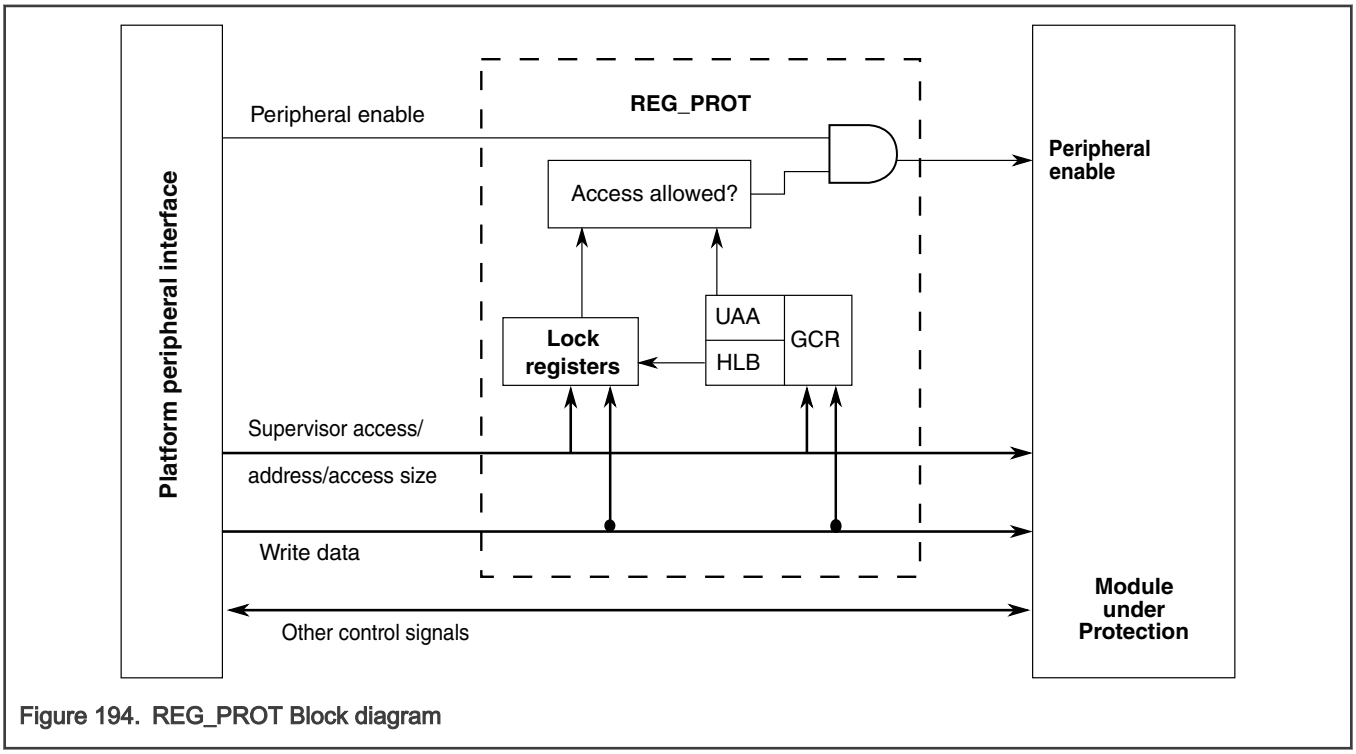


Figure 194. REG_PROT Block diagram

52.2.2 Features

REG_PROT includes these distinctive features:

- Write accesses for the module under protection can be restricted to the supervisor mode only.
- Registers of modules with their register slot size occupying 1 KB to 32 KB of memory-mapped address space, depending on the PROT_MEM parameter, can be dynamically locked. See Memory space and Protection size.
- Multiple ways are present to set the lock bits.
- After the lock bits are configured, they can be protected from changes.

52.3 Functional description

The following sections describe the functional characteristics of the REG_PROT module.

52.3.1 Modes of operation

The mode of REG_PROT depends on the Module Under Protection (MUP). The REG_PROT is operable when the MUP is operable.

The below table shows the write operations in user mode.

Table 249. User mode access table

GCR[UAA]	SLBRn[SLBm]	Write Operation on MUP	Write Operation on REG_PROT
0	0	Transfer error generated, write operation unsuccessful.	Transfer error generated, write operation unsuccessful.
0	1	Transfer error generated, write operation unsuccessful.	Transfer error generated, write operation unsuccessful.
1	0	• If register is implemented and write operation allowed on accessed address location then write operation is successful. • If register is not implemented or write operation is not allowed on accessed address location then transfer error is generated, write operation is unsuccessful.	• If register is implemented and write operation allowed on accessed address location then write operation is successful. • If register is not implemented or write operation is not allowed on accessed address location then transfer error is generated, write operation is unsuccessful.
1	1	Transfer error generated, write operation unsuccessful.	• If register is implemented and write operation allowed on accessed address location then write operation is successful. • If register is not implemented or write operation is not allowed on accessed address location then transfer error is generated.

Note:

- Read operation is always allowed
- n >= 0, m = [0,3]

52.3.2 Memory space and Protection size

This section provides a detailed description of the memory space and protection size of a module using REG_PROT.

The PROT_MEM size varies per chips. It can be:

- 1 KB
- 2 KB
- 4 KB
- 8 KB
- 16 KB
- 32 KB

The resulting memory space is divided into four areas. Memory locations within area #1 and area #2 are a part of the module under protection. Area #3 and area #4 are a part of REG_PROT.

The proper register slot size for each protected module is mentioned in the REG_PROT details file attached to this document.

Following is the memory space for a module that is protected by the REG_PROT module. Reserved registers in area #1 are handled as specified in the protected module's documentation. There is a single register in configuration space that corresponds

to area #4 while the size mentioned in Area size column is to turn the whole area multiple of 4 KB. See the Offset column to know the exact location of the GCR register.

Table 250. Area offset based on protection size

PROT_MEM	Area	Area size	Use	Offset
1 KB	Area 1	1 KB	Module Register(MR0—MR1023)	0000h—03FFh
	Area 2	1 KB	Module Register and Set Soft Lock Bit (LMR0—LMR1023)	0400h—07FFh
	Area 3	256 B	Soft Lock Bit Register (SLBR0—255)	0800h—08FFh
	Area 4	256 B	Global Configuration Register (GCR)	0900h—09FFh
		1.5 KB	Reserved	0A00h— 0FFFh
2 KB	Area 1	2 KB	Module Register (MR0—MR2047)	0000h—07FFh
	Area 2	2KB	Module Register and Set Soft Lock Bit (LMR0—LMR2047)	0800h—0FFFh
	Area 3	512 B	Soft Lock Bit Register (SLBR0—511)	1000h—11FFh
	Area 4	256 B	Global Configuration Register (GCR)	1200h—12FFh
		3.25 KB	Reserved	1300h—1FFFh
4 KB	Area 1	4 KB	Module Register (MR0—MR4095)	0000h—0FFFh
	Area 2	4 KB	Module Register and Set Soft Lock Bit (LMR0—LMR4095)	1000h—1FFFh
	Area 3	1 KB	Soft Lock Bit Register (SLBR0—1023)	2000h—23FFh
	Area 4	256 B	Global Configuration Register (GCR)	2400h—24FFh
		6.75 KB	Reserved	2500h—3FFFh
8 KB	Area 1	8 KB	Module Register (MR0—MR8191)	0000h—1FFFh
	Area 2	8 KB	Module Register and Set Soft Lock Bit (LMR0—LMR8191)	2000h—3FFFh
	Area 3	2 KB	Soft Lock Bit Register (SLBR0—2047)	4000h—47FFh
	Area 4	256 B	Global Configuration Register (GCR)	4800h—48FFh
		1.75 KB	Reserved	4900h—4FFFh
16 KB	Area 1	16 KB	Module Register (MR0—MR16383)	0000h—3FFFh
	Area 2	16 KB	Module Register and Set Soft Lock Bit (LMR0—LMR16383)	4000h—7FFFh
	Area 3	4 KB	Soft Lock Bit Register (SLBR0—4095)	8000h—8FFFh
	Area 4	4 KB	Global Configuration Register (GCR)	9000h—9FFFh
32 KB	Area 1	32 KB	Module Register (MR0—MR32767)	0000h—7FFFh
	Area 2	32 KB	Module Register and Set Soft Lock Bit (LMR0—LMR32767)	8000h—FFFFh
	Area 3	8 KB	Soft Lock Bit Register (SLBR0—8191)	10000h—11FFFh
	Area 4	4 KB	Global Configuration Register (GCR)	12000h—12FFFh

- Area #1 can take the memory space of 1 KB/2 KB/4 KB/8 KB/16 KB/32 KB, which holds the normal functional module registers and is transparent for all read/write operations.
- Area #2 can take the memory space of 1 KB/2 KB/4 KB/8 KB/16 KB/32 KB and is a mirror of area #1. For an example, if area #1 takes 8 KB of memory space, then read/write access to an offset 2000+X reads/writes the register at offset X. However, a write access to offset 2000+X additionally sets the optional soft lock bits for this offset X in the same cycle

as the register at offset X is written. This provides for an automatic write and lock operation. Not all registers in area #1 need to have protection defined by the associated soft lock bits. For unprotected registers at offset Y, accesses to offset 2000+Y are identical to accesses at offset Y.

- Area #3 can take the memory space of 256 B/512 B/1 KB/2 KB/4 KB/8 KB and hold soft lock bits, one bit per byte in area #1. The four soft lock bits associated with one module register word are arranged at byte boundaries in the memory map. The Soft Lock Bit registers can be directly written using a bit mask.
- Area #4 can take the memory space of 1.75 KB/3.5 KB/7 KB/2 KB/4 KB/4 KB and holds the configuration bits of the protection mode. There is one configuration Hard lock bit per module that prevents all further modifications to the soft lock bits and can only be cleared by a system reset after it is set. When user access allowed bit is set, it allows user access to the protected module.

If you access a locked byte with a write transaction, a bus error is issued to the system and the write transaction is not executed. This is true even if not all accessed bytes are locked.

Accessing unimplemented 32—bit registers in area #3 results in a bus transfer error. In area #4, there will not be any transfer error for the address range mentioned in the table above because these address spaces of area #4 are mapped to the GCR register. Accesses in area #4, beyond the address range mentioned in the Module memory map table, are prohibited and might or might not result into a bus abort.

52.3.2.1 Module register (MR)

Each functional module has its own unique set of registers. This chapter refers to these registers generically as module registers (MR), which exist in the lower module memory space (memory area 1), and receive protection from the REG_PROT module.

52.3.2.2 Module Register and Set Soft Lock Bit (LMR)

This is memory area #2 that provides mirrored access to module registers with the side-effect of setting soft lock bits in case of a write access to a register that is defined as protectable by the locking mechanism. Each MR byte is protectable by one associated bit in SLBR*n* [SLB*m*], according to the mapping described in Soft Lock Bit Register (SLBR*n*).

52.3.3 General

This module provides a generic register (address) write-protection mechanism. The register protection size can be:

- 32 bits (address == multiples of 4)
- 16 bits (address == multiples of 2)
- 8 bits (address == multiples of 1)

The addresses that are protected and the register protection size depends on the chip and/or module.

For all addresses that are protected, there are SLBR*n*[SLB*m*] bits that specify whether the address is locked. When an address is locked, it can only be read but not written in any mode (supervisor/normal). If an address is unprotected, the corresponding SLBR*n*[SLB*m*] bit is always 0, regardless of what the software writes to that field.

NOTE

For more information on register protection specification, see the REG_PROT details file attached to this document.

52.3.4 Change lock settings

To change the setting whether an address is locked or unlocked, the corresponding SLBR*n*[SLB*m*] bit needs to be changed. This can be done using the following methods:

- Set the SLBR*n*[SLB*m*] bit(s) by writing to area #2
- Modify the SLBR*n*[SLB*m*] directly by writing to area #3

Both methods are explained in the following sections.

52.3.4.1 Change lock settings via area #2

You can lock a register after writing to it. Note that reading area #2 does not affect soft lock bits. To do so, you must use area # 2.

The following shows an example with PROT_MEM = 8 KB where SLBs are modified using area #2.

When writing 16-bit to address 0008h, MR9 and MR8 in the protected module are updated. The corresponding lock fields remain unchanged (see the left part of Figure 195).

When writing 16-bit to address 2008h, MR9 and MR8 in the protected module are updated. The corresponding lock fields SLBR2[SLB0:1] become 1, and the lock fields SLBR2[SLB2:3] remain unchanged (see the right part of Figure 195).

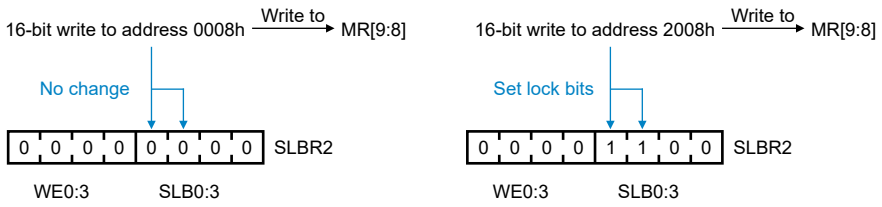


Figure 195. Enable locking via area #2

The following figure shows an example where some addresses are protected and some are not.

In Figure 196, addresses 000Ch and 000Dh are unprotected. Therefore, their corresponding lock fields SLBR3[SLB0:1] are always 0 (shown in bold). During a 32-bit write access to address 200Ch, the lock fields in SLBR3 change as follows:

- SLBR3[SLB2:3] become 1.
- SLBR3[SLB0:1] remain 0.

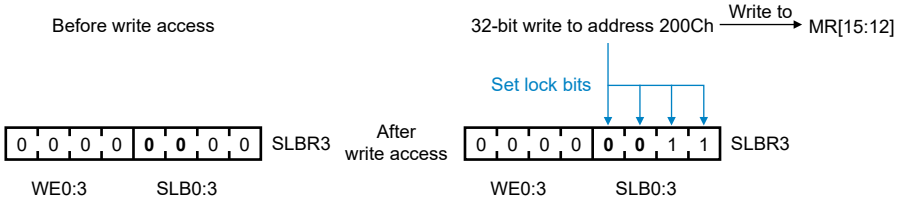


Figure 196. Enable locking for protected and unprotected addresses

52.3.4.2 Change lock settings via area #3

The lock bits are located in area #3 (see Table 250). You can modify them by writing to them. Each SLB*m* field in Soft Lock Bit Register (SLBR*n*) has a mask field, WE*m*, that protects SLB*m* from modification. This masking makes read-modify-write operations unnecessary.

Figure 197 shows two modification examples for registers with 8-bit protection. In part A, a write access to SLBR*n* specifies a mask value that allows modification of all SLBR*n*[SLB0:3] fields. Part B specifies a mask that allows modification only to SLBR*n*[SLB1:3].

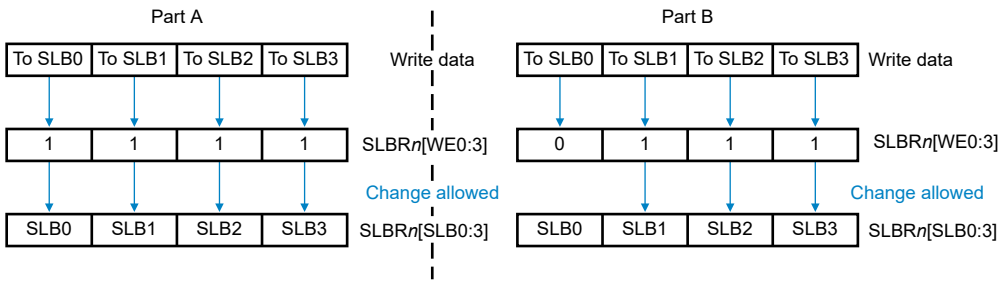


Figure 197. Change lock settings directly via area #3

Figure 198 shows examples for registers with 16-bit protection.

Part A of Figure 198 shows that for SLBR*n*:

- The data written to SLBR*n*[SLB0] is automatically written to SLBR*n*[SLB1].
- The data written to SLBR*n*[SLB2] is automatically written to SLBR*n*[SLB3].

This is because the address reflected by SLBR*n*[SLB0] and SLBR*n*[SLB2] are protected 16-bit wise. Note that in this case, the write enable SLBR*n*[WE0] and SLBR*n*[WE2] must be set while SLBR*n*[WE1] and SLBR*n*[WE3] don't matter.

Part B of Figure 198 shows that the data written to SLBR*n*[SLB0] is automatically written to SLBR*n*[SLB1]. This is done because the address reflected by SLBR*n*[SLB0] has 16-bit protection. In this case, SLBR*n*[WE0] must be 1 and SLBR*n*[WE1] does not matter. SLBR*n*[SLB2] and SLBR*n*[SLB3] remain unchanged because SLBR*n*[WE2] and SLBR*n*[WE3] are 0.

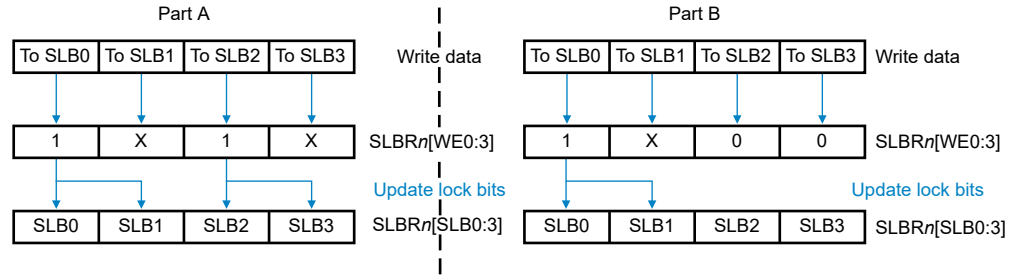


Figure 198. Change lock settings for addresses with 16-bit protection

Figure 199 shows a register with 32-bit protection. In SLBR*n*:

- When SLBR*n*[WE0]=1, the data written to SLBR*n*[SLB0] is automatically written to SLBR*n*[SLB1:3] as well.
- When SLBR*n*[WE0]=0, then SLBR*n*[SLB0:3] remain unchanged. Note that in this case, the write enable SLBR*n*[WE0] must be set while SLBR*n*[WE1:3] does not matter.

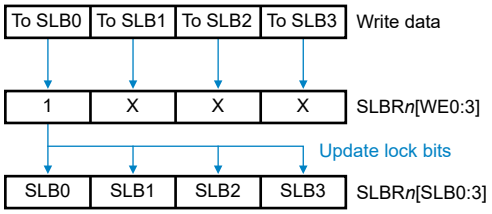


Figure 199. Change lock settings for addresses with 32-bit protection

The following figure shows a mixed protection size configuration.

In SLBR*n*:

- The data written to SLBR*n*[SLB0] is mirrored to SLBR*n*[SLB1] because the corresponding register has 16-bit protection.
- The data written to SLBR*n*[SLB2] is blocked because the corresponding register is unprotected.
- The data written to SLBR*n*[SLB3] is successfully written to SLBR*n*[SLB3] because the corresponding register has 8-bit protection.

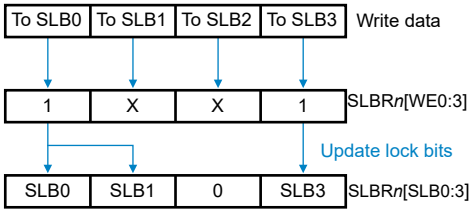


Figure 200. Change lock settings for mixed protection

52.3.4.3 Write protection for locking bits

Changing the locking bits through any of the procedures mentioned in Change lock settings via area #3 and Change lock settings via area #2 is only possible as long as the field GCR.HLB is cleared. After you write 1 to this field, the locking bits cannot be modified until there is a system reset.

52.3.5 Access errors

REG_PROT generates transfer errors under several circumstances, as described below. For the area definition, see Memory space and Protection size.

- If accessing area #1 or area #2, REG_PROT sends any access error from the underlying module under protection.
- If User mode is not allowed, the attempted User mode writes to all areas cause a transfer error, and the writes are blocked.
- If accessing the reserved area, a transfer error is asserted.
- If accessing unimplemented 32-bit registers in areas #3 and #4, a transfer error is asserted.
- If writing to a register in areas #1 and #2, and an SLB field is 1 for any of the affected bytes, then:
  - A transfer error is asserted.
  - The write is blocked.
  - The complete write operation to non-protected bytes in this word is ignored.
- If writing to an SLBR in area #3 with GCR[HLB]=1, a transfer error is asserted.
- Any write operation in any access mode to area #2, when GCR[HLB]=1, is not allowed.

NOTE

The transfer error generated by REG_PROT can only be observed on the CPU that initiates the target register write and has defined the strongly ordered memory region (through its MPU or MMU) covering the target register address.

52.4 External Signals

There are no external signals.

52.5 Initialization

The following sections contain information related to initialization of the Register Protection module.

52.5.1 Sequence

You must configure the REG_PROT first. And then, you need to write 1 to the HLB field in REG_PROT. This ensures that registers under protection are not updated in the user access mode. You need to follow these steps:

1. Enable XRDC protection for the module to enable writes in the supervisor mode only. This step is mandatory, if protected registers of the module are expected to be configured multiple times.
2. Configure REG_PROT to enable write on registers of module under protection.
3. Configure the registers of module under protection.
4. Configure the REG_PROT for the required SLB fields.
5. Set the HLB field in REG_PROT. This step is mandatory, if protected registers of module are expected to be configured only once.

52.5.2 Reset

It is a single clock IP, with single reset.

52.6 REG_PROT register descriptions

This register information documents memory area #3 and area #4.

52.6.1 REG_PROT memory map

REG_PROT base address: base address of protected module instance

Table 251. REG_PROT memory map

PROT_MEM	Offset	Register	Width (In bits)	Access	Reset value
1kB	0x800 – 0x8FF	Soft Lock Bit Register (SLBR0-255)	8	RW	00h
	0x900 – 0x9FF	Global Configuration Register (GCR)	32	RW	0000_0000h
2kB	0x1000 – 0x11FF	Soft Lock Bit Register (SLBR0-511)	8	RW	00h
	0x1200 – 0x12FF	Global Configuration Register (GCR)	32	RW	0000_0000h
4kB	0x2000 – 0x23FF	Soft Lock Bit Register (SLBR0-1023)	8	RW	00h
	0x2400 – 0x24FF	Global Configuration Register (GCR)	32	RW	0000_0000h
8kB	0x4000 – 0x47FF	Soft Lock Bit Register (SLBR0-2047)	8	RW	00h
	0x4800 – 0x48FF	Global Configuration Register (GCR)	32	RW	0000_0000h
16kB	0x8000 – 0x8FFF	Soft Lock Bit Register (SLBR0-4095)	8	RW	00h
	0x9000-0x9FFF	Global Configuration Register (GCR)	32	RW	0000_0000h
32kB	0x10000-0x11FFF	Soft Lock Bit Register (SLBR0-8191)	8	RW	00h
	0x12000 – 0x12FFF	Global Configuration Register (GCR)	32	RW	0000_0000h

52.6.2 Soft Lock Bit Register (SLBR*n*)

Offset

See REG_PROT memory map for SLBR*n* offset ranges. For SLBR*n* addresses, see the REG_PROT details file attached to this document.

Function

These registers hold the Soft Lock Bits (SLBs) for the protected registers in memory area #1, which is the normal register address space of the protected module. Multiple Soft Lock Bit Registers (SLBR*n*) can be implemented, depending on the number of protected module register bytes. Each SLBR*n* has four soft lock fields (SLB0-SLB3), each of which controls write access to a byte in memory area #1. Each soft lock field also has a corresponding write enable field in the same register that controls whether the soft lock field can be written. The following table shows the mapping between the soft lock fields to the bytes in memory area #1.

Table 252. SLBs vs. protected address

SLB	Protected address
SLBR0[SLB0]	MR0
SLBR0[SLB1]	MR1
SLBR0[SLB2]	MR2
SLBR0[SLB3]	MR3
SLBR1[SLB0]	MR4
SLBR1[SLB1]	MR5
SLBR1[SLB2]	MR6
SLBR1[SLB3]	MR7
SLBR2[SLB0]	MR8
...	...

Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0	0	0	SLB0	SLB1	SLB2	SLB3
W	WE0	WE1	WE2	WE3				
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 WE0	Write enable fields for SLB WE0 enables writing to SLB0. 0b - SLB is not modified. 1b - Value is written to SLB.
6 WE1	Write enable fields for SLB WE1 enables writing to SLB1. 0b - SLB is not modified. 1b - Value is written to SLB.
5	Write enable fields for SLB

Table continues on the next page...

Table continued from the previous page...

Field	Function
WE2	WE2 enables writing to SLB2. 0b - SLB is not modified. 1b - Value is written to SLB.
4 WE3	Write enable bits for SLB WE3 enables writing to SLB3. 0b - SLB is not modified. 1b - Value is written to SLB.
3 SLB0	Soft lock fields for one MRn register SLB0 can block accesses to MR[n *4 + 0]. 0b - Associated MRn byte is unprotected and writable. 1b - Associated MRn byte is locked against write accesses.
2 SLB1	Soft lock bits for one MRn register SLB1 can block accesses to MR[n *4 + 1]. 0b - Associated MRn byte is unprotected and writable. 1b - Associated MRn byte is locked against write accesses.
1 SLB2	Soft lock fields for one MRn register SLB2 can block accesses to MR[n *4 + 2]. 0b - Associated MRn byte is unprotected and writable. 1b - Associated MRn byte is locked against write accesses.
0 SLB3	Soft lock fields for one MRn register SLB3 can block accesses to MR[n *4 + 3]. 0b - Associated MRn byte is unprotected and writable. 1b - Associated MRn byte is locked against write accesses.

52.6.3 Global Configuration Register (GCR)

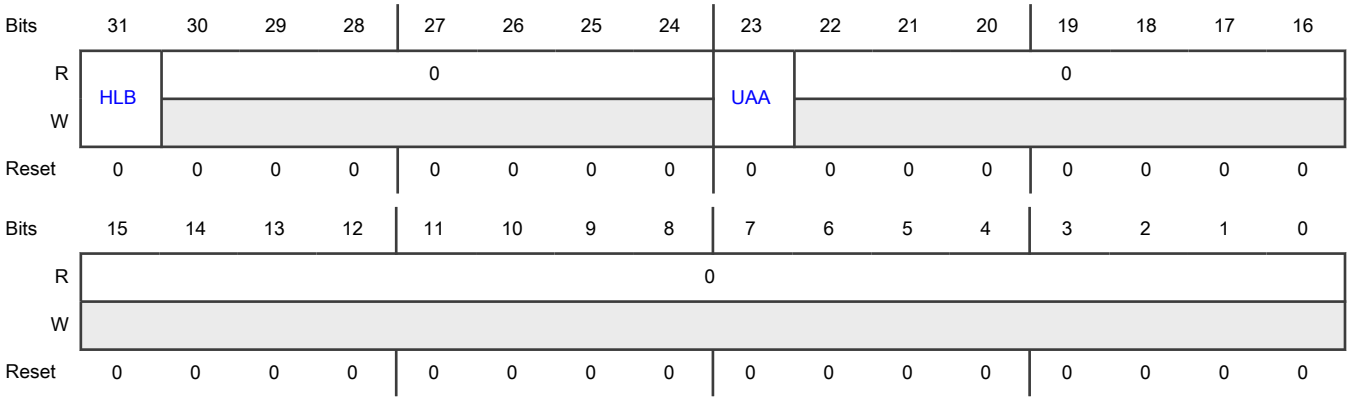
Offset

See REG_PROT memory map for GCR offset.

Function

This register controls module level configurations.

Diagram



Fields

Field	Function
31 HLB	Hard Lock Bit This field cannot be cleared after it is set by the software. It can only be cleared by a system reset. 0b - All SLB fields are accessible and can be modified. 1b - All SLB fields are write protected and cannot be modified.
30-24 —	Reserved
23 UAA	User Access Allowed Controls the user and supervisor mode access to registers of the module under protection. 0b - The registers in the module under protection can only be written in the supervisor mode (user access is not allowed). All the write accesses in user (non-supervisor) mode are not executed and a transfer error is issued. This access restriction is in addition to any access restrictions imposed by the protected IP module. 1b - The registers in the module under protection can be accessed in the mode defined for the module registers without any additional restrictions. It can be modified in both user (non-supervisor) and supervisory mode.
22-0 —	Reserved

52.7 Glossary

PROT_MEM Integration parameter that specifies the size of the module register slot protected. The PROT_MEM size depends on the chip or module.

Hard lock A lock that restricts access to any register bit. You can unlock or clear this lock through the hardware reset.

Soft lock A lock that restricts access to any register bit. You can unlock or clear this lock through the software.

Chapter 53
Clock Monitoring Unit – Frequency Check (CMU_FC)

53.1 Chip-specific CMU_FC information

53.1.1 CMU_FC configuration

The device consists of a robust clock monitoring architecture targeted to monitor various clock nodes to ensure device clock robustness. There are upto seven clock monitoring units present in the device out of which 4 are of type CMU_FC. See 'Clock monitoring' section in Clocking chapter for details on device clock monitoring architecture.

NOTE

CMU_5 is accessible by Cortex-M0+ core only.

CMU_FC combined spec variation: The combined spec variation for CMU (used for configuring CMU_FCx.HTCR[HFREF] and CMU_FCx.HTCR[LFREF]) is the absolute sum of frequency variations of the clock sources. Refer device datasheet for frequency variations for the corresponding clock sources.

53.2 Introduction

CMU_FC ensures the clock integrity of selected clocks and allows continuous clock integrity verification.

CMU_FC checks if the frequency of a monitored clock (*monitored_clock*) is within a programmable frequency range specified by the user. If the frequency is outside the specified limits, it could lead to performance, protocol, and timing failures. CMU_FC requires a clock used as a base reference for the frequency check operation. This reference clock (*reference_clock*) must be within documented limits for correct CMU_FC operation.

53.2.1 Basic operation

CMU_FC counts clock cycles of the monitored clock during a user programmable time duration of *n* clock cycles of the reference clock. There is a comparison between the final count of the monitored clock and the user programmable upper and lower threshold count limits. The control logic determines whether the cycle count is within the programmed limits. If not it sets the Frequency Higher than High frequency reference threshold event status (FHH) field or the Frequency Lower than Low frequency reference threshold event status (FLL) field in the Status Register (SR). If you enable interrupts, an interrupt asserts when the status flag sets.

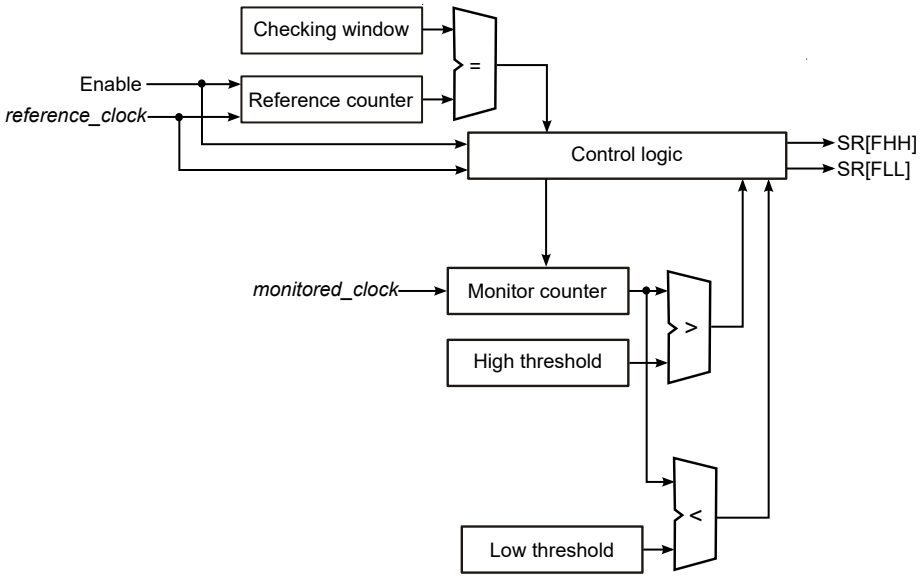


Figure 201. CMU_FC block diagram

CMU_FC cannot detect the following:

- Monitored clock duty cycle variations
- Instantaneous monitored clock frequency variations that do not cross upper or lower threshold limits

53.2.2 Features

The CMU_FC features are:

- Programmable duration of reference clock cycles
- Initiates an event if a monitored clock frequency is higher than high frequency reference (FHH)
- Initiates an event if a monitored clock frequency is lower than low frequency reference (FLL)
- Timeout functionality generates an FLL event if a monitored clock stops functioning
- Masking of FHH event interrupt
- Masking of FLL event interrupt

Table 253. Clock description

Clock	I/O	Description
reference_clock	I	Clock signal that the module uses as a reference to evaluate *monitored_clock*
monitored_clock	I	Clock signal on which frequency check is performed
bus_clock	I	Clock signal on which register read/write operation is performed

NOTE

See the Clocking chapter for details of monitored and reference clocks used on this chip.

53.3 Functional description

53.3.1 Frequency checking

Frequency checking starts when software writes GCR[FCE] = 1. The Run Status field SR[RS] shows 1 after frequency check operation starts.

CMU_FC performs continuous periodic clock checking of the monitored clock. Each checking window involves the following operations:

- Stage 1 – Reference clock counter runs for RCCR[REF_CNT] reference clock cycles. Monitored clock counter runs in parallel for the same time duration as reference clock counter.
- Stage 2 – After stage 1, the module compares the final monitored clock count against the programmed thresholds.

If the monitored clock count is greater than the high threshold value in HTCR[HFREF], an FHH event occurs.

If the monitored clock count is lower than the low threshold value in LTCR[LFREF], an FLL event occurs.

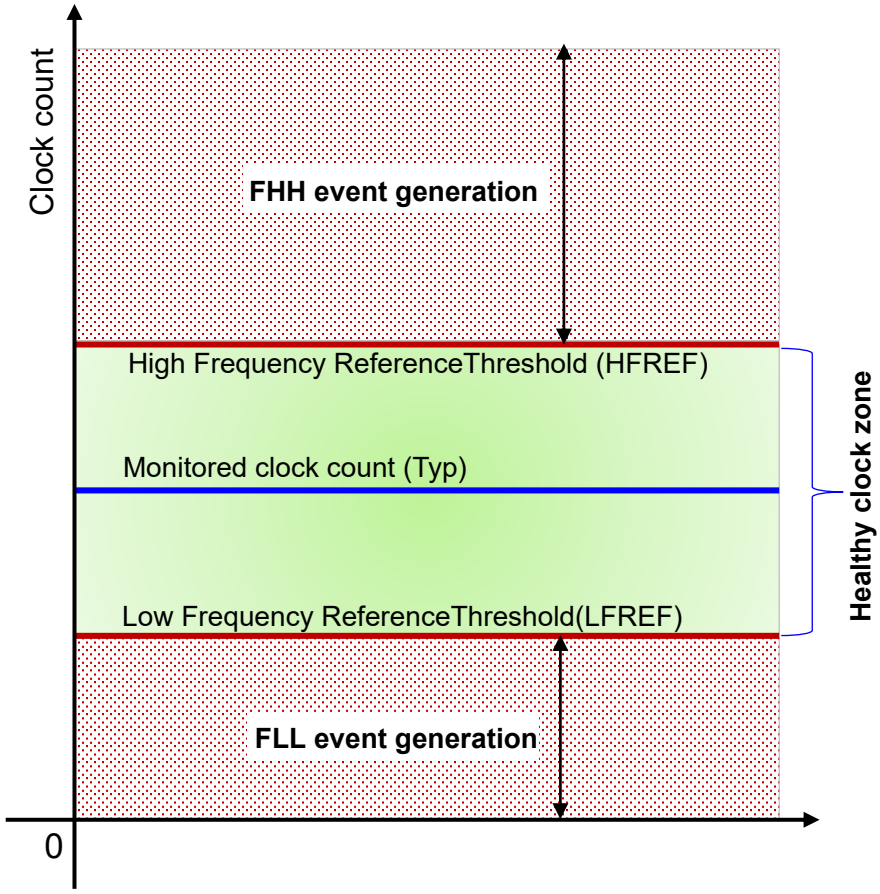


Figure 202. Frequency checking HFREF and LFREF thresholds

53.3.1.1 Monitored clock lost

If a monitored clock ceases operation before evaluation of that clock (Stage 2, see Frequency checking), CMU_FC waits for RCCR[REF_CNT] reference clock periods by extending Stage 2 before triggering an FLL event.

- If *monitored_clock* is not running when enabling CMU_FC:

Worst case FLL response time = (2 × RCCR[REF_CNT] + 12) *reference_clock* cycles + 15 *bus_clock* cycles

- If *monitored_clock* stops when the CMU_FC is running:

Worst case FLL response time = (2 × RCCR[REF_CNT] + 2) *reference_clock* cycles + 3 *bus_clock* cycles

53.4 Programming guidelines

53.4.1 Programming RCCR[REF_CNT]

RCCR[REF_CNT] defines the duration of the checking operation in number of *reference_clock* cycles. The minimum RCCR[REF_CNT] value can be calculated using the following formula:

Minimum RCCR [REF_CNT] = CEILING value of MAX $\left(3 \text{ X } \left(\frac{f_{reference_clock}}{f_{bus_clock}}\right), 8 + 5 \text{ X } \left(\frac{f_{reference_clock}}{f_{monitored_clock}}\right)\right)$

...where:

- $f_{reference_clock}$ is the frequency of the reference clock
- f_{bus_clock} is the frequency of the bus clock
- $f_{monitored_clock}$ is the frequency of the monitored clock

Example to determine RCCR[REF_CNT]:

- f_{bus_clock} = 16 MHz
- $f_{reference_clock}$ = 8 MHz
- $f_{monitored_clock}$ = 48 MHz

Formula #1:	$3 \text{ X } \left(\frac{f_{reference_clock}}{f_{bus_clock}}\right)$
Inserting the values:	3 × (8 / 16) = 1.5
Formula #2:	$8 + 5 \text{ X } \left(\frac{f_{reference_clock}}{f_{monitored_clock}}\right)$
Inserting the values:	8 + 5 × (8 / 48) ≈ 8.83
MAX (1.5, 8.83):	8.83
CEILING value of MAX:	9
Therefore, minimum RCCR[REF_CNT]:	9

Programmed RCCR[REF_CNT] value must be greater than or equal to the calculated minimum value and should be decided after considering application requirements of frequency check completion response time and overall accuracy required from CMU_FC.

- Higher values of RCCR[REF_CNT] results in longer measurement window, leading to better accuracy in monitored clock check.
- Lower values of RCCR[REF_CNT] results in shorter measurement window, leading to faster FHH and FLL event response, but higher inaccuracy in reported result.

For measurement window calculation, see step 8 in Programming HFREF and LFREF

53.4.2 Programming HFREF and LFREF

You must consider the following when programming HFREF/LFREF:

- Tolerable monitored clock frequency variation
- Maximum reference clock frequency variation
- Inherent module inaccuracy

CMU_FC has an expected maximum deviation of ± 3 *monitored_clock* cycles (CMU_FC_{VAR+} = 3, CMU_FC_{VAR-} = –3).

The following procedure shows how to calculate optimum values for HTCR[HFREF] and LTCR[LFREF] (numbers shown are from example in table):

1. Determine the ideal *monitored_clock* frequency ($f_{monitored_clock}$ (ideal), 48 MHz).
2. Determine the specified variation of the *monitored_clock* (1.1%).
3. Calculate *monitored_clock* frequency variation: .

$$f_{monitored_clock}\ (ideal)\ \text{x}\ (1 \pm variation\ (\text{step 2}))$$

   - $f_{monitored_clock}$ (max) = 48 MHz × (1 + (1.1 ÷ 100)) = 48.53 MHz
   - $f_{monitored_clock}$ (min)= 48 MHz × (1 – (1.1 ÷ 100)) = 47.47 MHz
4. Determine the ideal *reference_clock* frequency ($f_{reference_clock}$ (ideal), 8 MHz).
5. Determine the specified variation of the *reference_clock* (3.3%).
6. Calculate ideal *reference_clock* frequency variation:

$$f_{reference_clock}\ (ideal)\ \text{x}\ (1 \pm variation\ (\text{step 5}))$$

   - $f_{reference_clock}$ (max) = 8 MHz × (1 + (3.3 ÷ 100)) = 8.26 MHz
   - $f_{reference_clock}$ (min) = 8 MHz × (1 – (3.3 ÷ 100)) = 7.74 MHz
7. Select the value of RCCR[REF_CNT] needed (80). See Programming RCCR[REF_CNT].
8. Calculate ideal measurement window:

$$\frac{RCCR\ [REF_CNT]}{f_{reference_clock}\ (ideal)}$$

   - 80 ÷ 8 MHz = 10000 ns
9. Calculate ideal *monitored_clock* count:

$$\frac{RCCR[REF_CNT]}{f_{reference_clock}(ideal)} \times f_{monitored_clock}\ (ideal)$$

   - 80 ÷ 8 MHz × 48 MHz = 480.00 (round off this value, 480 in this case)
10. Calculate high threshold value (HTCR[HFREF]):

$$\text{Ceiling value of } \left(\frac{f_{monitored_clock}\,(max)}{f_{reference_clock}\,(min)} \times RCCR[REF_CNT]\right) + CMU_FC_{VAR+}$$

- HTCR[HFREF] = Ceiling(48.53 MHz ÷ 7.74 MHz × 80 + 3) = 505

11. Calculate low threshold value (LTCR[LFREF]):

$$\text{Floor value of } \left(\frac{f_{monitored_clock}\,(min)}{f_{reference_clock}\,(max)} \times RCCR[REF_CNT]\right) - CMU_FC_{VAR-}$$

- LTCR[LFREF] = Floor(47.47 MHz ÷ 8.26 MHz × 80 – 3) = 456

Table 254. HTCR[HFREF] and LTCR[LFREF] calculation example

Property	Value	Unit
Bus clock frequency (f_{bus_clock})	16	MHz
Monitored clock frequency ($f_{monitored_clock}$ (ideal))	48	MHz
Specified variation of monitored clock	1.1	%
Monitored clock frequency (maximum after specified variation, $f_{monitored_clock}$ (max))	48.53	MHz
Monitored clock frequency (minimum after specified variation, $f_{monitored_clock}$ (min))	47.47	MHz
Reference clock frequency ($f_{reference_clock}$ (ideal))	8	MHz
Specified variation of reference clock	3.3	%
Reference clock frequency (maximum after specified variation, $f_{reference_clock}$ (max))	8.26	MHz
Reference clock frequency (minimum after specified variation, $f_{reference_clock}$ (min))	7.74	MHz
Minimum reference count (RCCR[REF_CNT])	9	—
Programmed reference count (RCCR[REF_CNT]) (must be ≥ minimum RCCR[REF_CNT])	80	—
Measurement window	10000.00	ns
Calculated monitored clock count	480.00	—
Calculated monitored clock Count (rounded)	480	—
Module positive variation (CMU_FC_{VAR+})	3	—
Module negative variation (CMU_FC_{VAR-})	-3	—
Programmed higher threshold (HTCR[HFREF])	505	—

Table continues on the next page...

Table 254. HTCR[HFREF] and LTCR[LFREF] calculation example (continued)

Property	Value	Unit
Programmed lower threshold (LTCR[LFREF])	456	—

53.4.3 Module programming sequence

The recommended programming sequence for CMU_FCfrequency checking is:

1. In any order, complete the following:
   - Program RCCR[REF_CNT] to define the frequency measuring window duration in *reference_clock* cycles.
   - Program LTCR[LFREF] and HTCR[HFREF] to set the permissible frequency range of the *monitored_clock*.
   - Program IER to enable interrupt.
2. Write GCR[FCE] = 1 to start the frequency check operation.

   NOTE

   When GCR[FCE] = 1, you must not write to RCCR, HTCR, LTCR, or IER. Attempting to do so generates a bus transfer error.
3. To reconfigure RCCR, LTCR, HTCR , and IER or to stop frequency check operation, wait for SR[RS] = 1, then write GCR[FCE] = 0.

53.5 CMU_FC register descriptions

53.5.1 CMU_FC memory map

This section describes the address order of all the CMU_FC registers. Each description includes a standard register diagram and associated field descriptions.

CMU_0 base address: 402B_C000h

CMU_3 base address: 402B_C060h

CMU_4 base address: 402B_C080h

CMU_5 base address: 402B_C0A0h

Offset	Register	Width (In bits)	Access	Reset value
0h	Global Configuration Register (GCR)	32	RW	0000_0000h
4h	Reference Count Configuration Register (RCCR)	32	RW	0000_0000h
8h	High Threshold Configuration Register (HTCR)	32	RW	00FF_FFFFh
Ch	Low Threshold Configuration Register (LTCR)	32	RW	0000_0000h
10h	Status Register (SR)	32	RW	0000_0000h
14h	Interrupt Enable Register (IER)	32	RW	0000_0000h

53.5.2 Global Configuration Register (GCR)

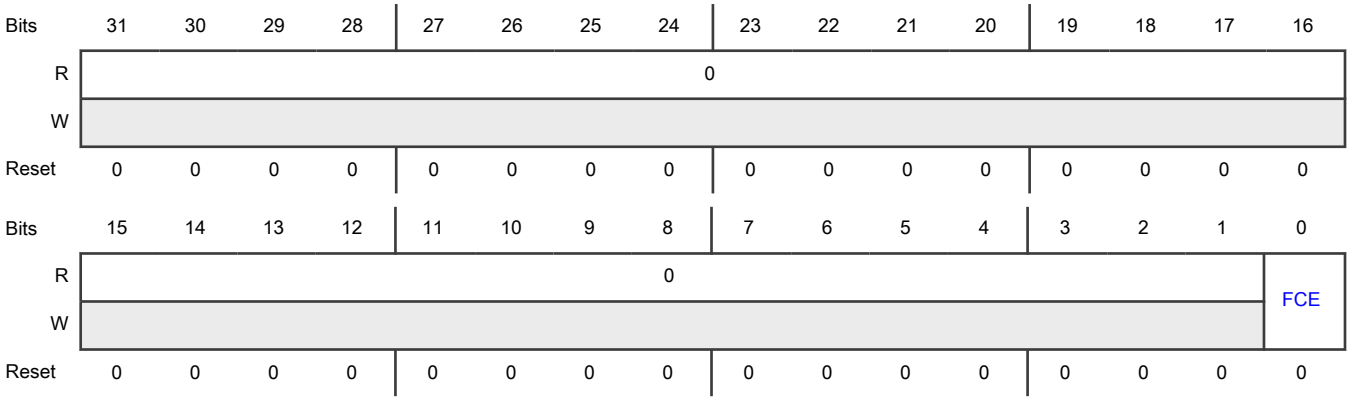
Offset

Register	Offset
GCR	0h

Function

Controls module level configurations such as enabling frequency check.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 FCE	Frequency Check Enable Starts or stops frequency checking. FCE is disabled by default. Software can enable FCE at any time. To stop the ongoing operation, write 0 to FCE only when SR[RS] = 1. 0b - Stops frequency checking 1b - Starts frequency checking

53.5.3 Reference Count Configuration Register (RCCR)

Offset

Register	Offset
RCCR	4h

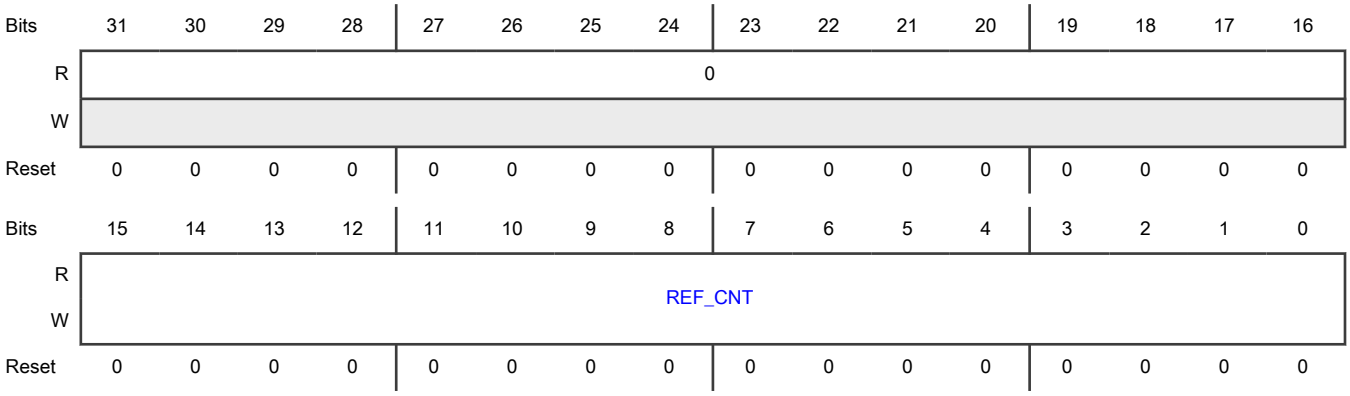
Function

Programs reference count duration of the frequency check window.

NOTE

Write to RCCR only when GCR[FCE] = 0. A bus transfer error results if software writes RCCR when GCR[FCE] = 1.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 REF_CNT	Reference clock count Total number of counts of *reference_clock* for which frequency check runs. This field defines the duration of one frequency check window. See Programming RCCR[REF_CNT] for RCCR calculation.

53.5.4 High Threshold Configuration Register (HTCR)

Offset

Register	Offset
HTCR	8h

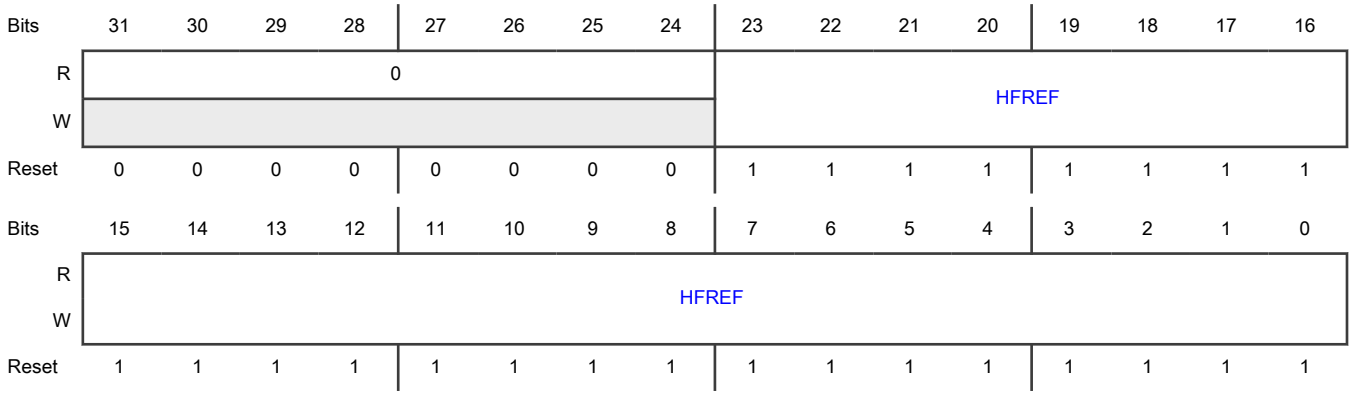
Function

Determines the high threshold limit of the monitored clock counter.

NOTE

Write HTCR only when GCR[FCE] = 0. A bus transfer error results if software writes HTCR when GCR[FCE] = 1.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 HFREF	High frequency reference threshold HFREF determines the high reference value for the monitored clock frequency. See Programming HFREF and LFREF for HFREF calculation. **NOTE** Do not program HFREF to a value greater than 0x00FFFFFC.

53.5.5 Low Threshold Configuration Register (LTCR)

Offset

Register	Offset
LTCR	Ch

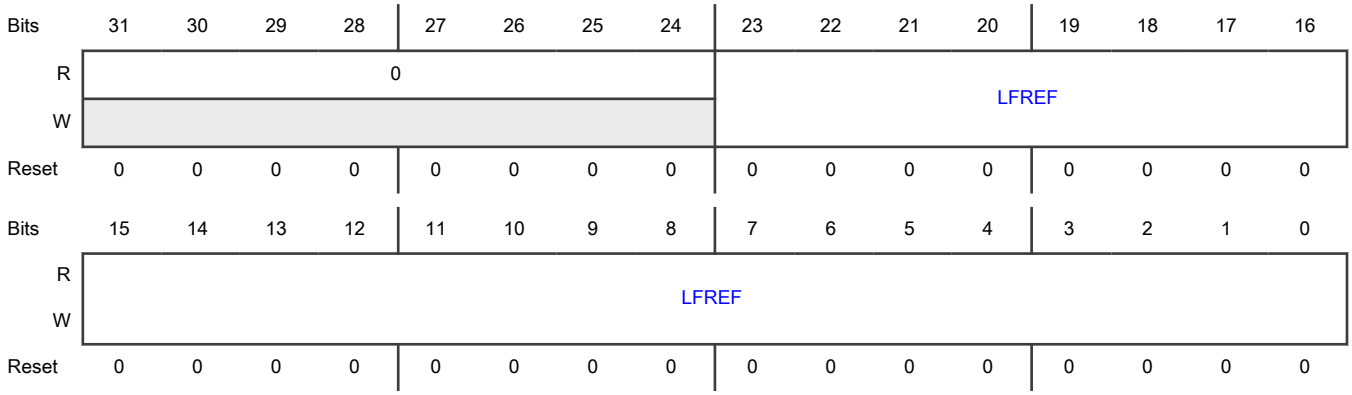
Function

Determines the low threshold limit of the monitored clock counter.

NOTE

Write LTCR only when GCR[FCE] = 0. A bus transfer error results if software writes LTCR when GCR[FCE] = 1.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 LFREF	Low Frequency Reference Threshold LFREF determines the low reference value for the monitored clock frequency. See Programming HFREF and LFREF for LFREF calculation. **NOTE** Do not program LFREF to a value less than 0x00000003.

53.5.6 Status Register (SR)

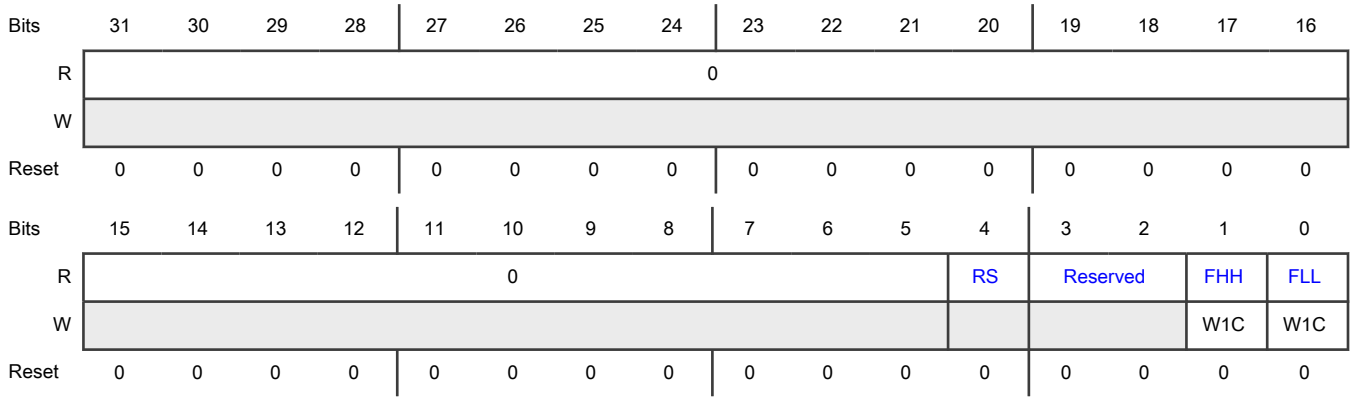
Offset

Register	Offset
SR	10h

Function

Provides the internal status of the module.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 RS	Run Status Shows the running status of module internal operation. After enabling the frequency check operation (GCR[FCE] = 1), there is a fixed delay before this field shows running status. If the system wants to disable the frequency check operation, then the software should read this register and write GCR[FCE] = 0 only when this field is 1. 0b - Frequency check stopped 1b - Frequency check running
3-2 —	Reserved
1 FHH	Frequency higher than high frequency reference threshold event status Hardware writes 1 to FHH when the monitored clock frequency becomes higher than the high threshold value in HTCR[HFREF]. FHH clears when software writes 1. 0b - No FHH event 1b - FHH event occurred
0 FLL	Frequency lower than low frequency reference threshold event status Hardware writes 1 to FLL when the monitored clock frequency becomes lower than the low threshold value in LTCR[LFREF]. FLL clears when software writes 1. 0b - No FLL event 1b - FLL event occurred

53.5.7 Interrupt Enable Register (IER)

Offset

Register	Offset
IER	14h

Function

Enables CMU_FC interrupts.

NOTE

Write IER only when GCR[FCE] = 0. A bus transfer error results if software writes IER when GCR[FCE] = 1.

Enable only either asynchronous FHH event interrupt or synchronous FHH event interrupt at a time.

For example:

- If IER[FHHIE] is set to 1 then set IER[FHHAIE] to 0.
- If IER[FHHAIE] is set to 1 then set IER[FHHIE] to 0.

Enable only either asynchronous FLL event interrupt or synchronous FLL event interrupt at a time.

For example:

- If IER[FLLIE] is set to 1 then set IER[FLLAIE] to 0.
- If IER[FLLAIE] is set to 1 then set IER[FLLIE] to 0.

See Clocking chapter for interrupt usage information based on CMU_FC instances.

Diagram

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

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

Fields

Field	Function
31-24 —	Reserved
23	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-4 —	Reserved
3 FHHAIE	Frequency Higher than High Frequency Reference Threshold Asynchronous Interrupt Enable FHHAIE enables FHH asynchronous interrupt at the module boundary. 0b - Asynchronous FHH event interrupt disabled 1b - Asynchronous FHH event interrupt enabled
2 FLLAIE	Frequency Lower than Low Frequency Reference Threshold Asynchronous Interrupt Enable FLLAIE enables FLL asynchronous interrupt at the module boundary. 0b - Asynchronous FLL event interrupt disabled 1b - Asynchronous FLL event interrupt enabled
1 FHHIE	Frequency Higher than High Frequency Reference Threshold Synchronous Interrupt Enable FHHIE enables an FHH synchronous interrupt at the module boundary. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
CMU_0	IER	—
CMU_3	—	IER
CMU_4	—	IER

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>CMU_5</td><td>—</td><td>IER</td></tr>
</table>
0b - Synchronous FHH event interrupt disabled
1b - Synchronous FHH event interrupt enabled
</td></tr>
<tr><td>0
FLLIE</td><td>Frequency Lower than Low Frequency Reference Threshold Synchronous Interrupt Enable
FLLIE enables an FLL synchronous interrupt at the module boundary.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>CMU_0</td><td>IER</td><td>—</td></tr>
<tr><td>CMU_3</td><td>—</td><td>IER</td></tr>
<tr><td>CMU_4</td><td>—</td><td>IER</td></tr>
<tr><td>CMU_5</td><td>—</td><td>IER</td></tr>
</table>
0b - Synchronous FLL event interrupt disabled
1b - Synchronous FLL event interrupt enabled
</td></tr>
</table>

53.6 Glossary

FHH The state that the monitored clock frequency is higher than the high frequency reference. FHH is also the name of the register field that indicates this state.

FHH event The event that the module triggers when it detects FHH.

FLL The state that the monitored clock frequency is lower than the lower frequency reference. FLL is also the name of the register field that indicates this state.

FLL event The event that the module triggers when it detects FLL.

Chapter 54
Clock Monitoring Unit – Frequency Meter (CMU_FM)

54.1 Chip-specific CMU_FM information

54.1.1 CMU_FM configuration

The device consists of a robust clock monitoring architecture targeted to monitor various clock nodes to ensure device clock robustness. There are upto seven clock monitoring units present in the device out of which 2 are of type CMU_FM. See 'Clock monitoring' section in Clocking chapter for details on device clock monitoring architecture.

NOTE

CMU metering interrupt is not generated if the clock which is being metered is lost. A timeout status flag, CMU.SR[FMTO] is updated in this case.

NOTE

Software must ensure that CMU1 interrupt is serviced within POR_WDG timeout. In absence of CMU1 interrupt servicing, the POR_WDG detects this as an FIRC failure and initiates a POR recovery.

54.2 Introduction

CMU_FM measures the frequency of the metered clock (*metered_clock*) using another clock as reference. This reference clock (*reference_clock*) must be within documented limits for correct CMU_FM operation. CMU_FM operation is only guaranteed if a reference clock is within normal operating parameters.

54.2.1 Basic operation

The CMU_FM counts the clock cycles of a metered clock in the user programmable time duration *n* number of clock cycles of the reference clock. After completion of a metered clock operation, a register field sets, indicating the operation has completed. The count of the metered clock also updates in the Status Register (SR) after completion of a metered clock operation.

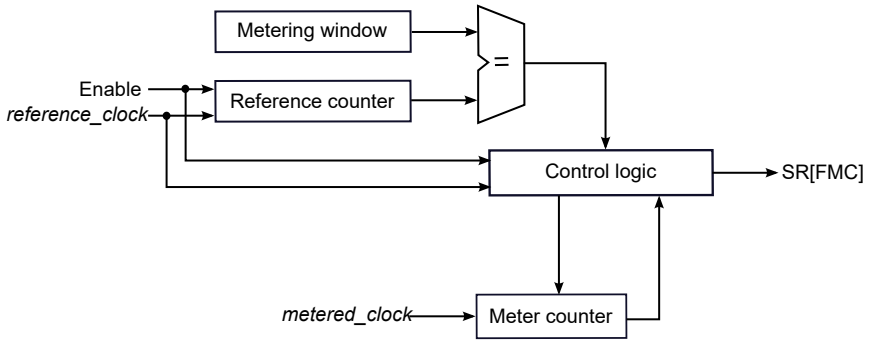


Figure 203. Block diagram

54.2.2 Features

Features of the CMU_FM are:

- Programmable duration of *reference_clock* cycles
- Maskable interrupt generation when frequency metering completes
- Timeout feature indicates loss of *metered_clock*

Table 255. Clock description

Clock	I/O	Description
reference_clock	I	Clock signal that the module uses as a reference to evaluate *metered_clock*
metered_clock	I	Clock signal on which frequency metering is performed
bus_clock	I	Clock signal on which register read/write operation is performed

NOTE

See the Clocking chapter for details of metered and reference clocks used on this chip.

54.3 Functional description

The frequency meter operation initiates when software writes GCR[FME] = 1. When the operation starts, hardware writes SR[RS] = 1. Safety critical applications must poll SR[RS] to determine when metering operation starts.

Frequency metering window involves these operations:

1. Stage 1 — The reference clock counter runs for RCCR[REF_CNT] cycles of *reference_clock*. The metered clock counter runs in parallel for the same time duration as the *reference_clock* counter.
2. Stage 2 — At the end of stage 1, SR[FMC] = 1, SR[MET_CNT] updates with the *metered_clock* count, and CMU_FM writes GCR[FME] = 0.

54.4 Programming guidelines

54.4.1 Programming RCCR[REF_CNT]

RCCR[REF_CNT] defines the duration of the metering operation in number of *reference_clock* cycles. The minimum RCCR[REF_CNT] value can be calculated using the following formula:

$$\textbf{Minimum RCCR [REF_CNT]} = \text{CEILING value of MAX}\left(3 \text{ X } \left(\frac{f_{reference_clock}}{f_{bus_clock}}\right), 8 + 5 \text{ X } \left(\frac{f_{reference_clock}}{f_{monitored_clock}}\right)\right)$$

...where:

- $f_{reference_clock}$ is the frequency of the reference clock
- f_{bus_clock} is the frequency of the bus clock
- $f_{metering_clock}$ is the frequency of the metered clock

Example to determine RCCR[REF_CNT]:

- f_{bus_clock} = 16 MHz
- $f_{reference_clock}$ = 8 MHz
- $f_{metered_clock}$ = 48 MHz

Formula #1:	$3 \text{ X } \left(\frac{f_{reference_clock}}{f_{bus_clock}} \right)$
Inserting the values:	3 × (8 / 16) = 1.5
Formula #2:	$6 + 4 \text{ X } \left(\frac{f_{reference_clock}}{f_{metered_clock}} \right)$
Inserting the values:	6 + 4 × (8 / 48) ≈ 6.67
MAX (1.5, 6.67):	6.67
CEILING value of MAX:	7
Therefore, minimum RCCR[REF_CNT]:	7

Programmed RCCR[REF_CNT] value must be greater than or equal to the calculated minimum value and should be decided after considering application requirements of frequency metering completion response time and overall accuracy required from CMU_FM.

- Higher values of RCCR[REF_CNT] results in longer metering window, leading to better accuracy in metered clock measurement.
- Lower values of RCCR[REF_CNT] results in shorter metering window, leading to faster FMC response, but higher inaccuracy in reported result.

54.4.2 Module programming sequence

The recommended programming sequence of CMU_FM is:

1. In any order, complete the following:
   - Program RCCR[REF_CNT] to define the frequency metering window duration in *reference_clock* cycles.
   - Program IER[FMCIE] to enable interrupt.
2. Write GCR[FME] = 1 to start frequency metering operation. Writes to RCCR and IER are disabled after GCR[FME] = 1. Attempting a write to any of them after GCR[FME] = 1 generates a bus transfer error.
3. To reconfigure RCCR and IER or to stop frequency metering operation, wait for SR[RS] = 1, then write GCR[FME] = 0.

Upon completion of metering operations:

- Hardware writes SR[FMC] = 1
- SR[MET_CNT] updates with metered count value
- Hardware writes GCR[FME] = 0

54.5 CMU_FM register descriptions

54.5.1 CMU_FM memory map

This section describes the address order of all the CMU_FM registers. Each description includes a standard register diagram and associated field descriptions.

CMU_1 base address: 402B_C020h

CMU_2 base address: 402B_C040h

Offset	Register	Width (In bits)	Access	Reset value
0h	Global Configuration Register (GCR)	32	RW	0000_0000h
4h	Reference Count Configuration Register (RCCR)	32	RW	0000_0000h
8h	Status Register (SR)	32	RW	0000_0000h
Ch	Interrupt Enable Register (IER)	32	RW	0000_0000h

54.5.2 Global Configuration Register (GCR)

Offset

Register	Offset
GCR	0h

Function
Controls module level configurations such as enabling frequency metering.

Diagram

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

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

Fields

Field	Function
31-1 —	Reserved
0 FME	Frequency Meter Enable Starts or stops frequency metering. FME is disabled by default. Software can enable FME at any time. To stop the ongoing operation, write 0 to FME only when SR[RS] = 1. FME automatically clears upon frequency metering completion or timeout event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Stops frequency metering 1b - Starts frequency metering

54.5.3 Reference Count Configuration Register (RCCR)

Offset

Register	Offset
RCCR	4h

Function

Programs reference count duration of the frequency meter window.

NOTE

Write to RCCR only when GCR[FME] = 0. A bus transfer error results if software writes RCCR when GCR[FME] = 1.

Diagram

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

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

Fields

Field	Function
31-16 —	Reserved
15-0 REF_CNT	Reference Clock Count Total number of counts of *reference_clock* for which frequency metering runs. This field defines the duration of one frequency metering window. See Programming RCCR[REF_CNT] for RCCR calculation.

54.5.4 Status Register (SR)

Offset

Register	Offset
SR	8h

Function

Provides the internal status of the module and metered clock count.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MET_CNT								0			RS	Reserved		FMTO	FMC
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 MET_CNT	Meter Clock Count This field stores the count value of the metered clock cycles when frequency metering operation is complete, in other words, SR[FMC] = 1. CMU_FM has a maximum deviation of ± 3 *metered_clock* cycles over the ideal *metered_clock* cycles expected by the user.
7-5 —	Reserved
4 RS	Run Status Shows the running status of module internal operation. After enabling the frequency metering operation (GCR[FME] = 1), there is a fixed delay before this field shows running status. If the system wants to disable the frequency metering operation, then the software should read this register and write GCR[FME] = 0 only when this field is 1. 0b - Frequency meter stopped 1b - Frequency meter running
3-2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 FMTO	Frequency Meter Time Out Sets if there is a loss of metered clock during the metering operation. After metering window, the module waits for a timeout duration of RCCR[REF_CNT] cycles before writing one to this field. GCR[FME] = 0 and SR[FMC] = 0 when FMTO = 1. Writing one clears this field.
0 FMC	Frequency Meter Operation Complete On completion of frequency metering operation, FMC = 1, SR[MET_CNT] field updates, and GCR[FME] clears. Writing one clears this field.

54.5.5 Interrupt Enable Register (IER)

Offset

Register	Offset
IER	Ch

Function

Enables CMU_FM interrupts.

NOTE

Write IER only when GCR[FME] = 0. A bus transfer error results if software writes IER when GCR[FME] = 1.

Diagram

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

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

Fields

Field	Function
31-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 FMCIE	Frequency Meter Complete Interrupt Enable This bit is used to enable/disable Frequency Meter complete interrupt at the module boundary. 0b - Frequency Meter complete interrupt is disabled 1b - Frequency Meter complete interrupt is enabled

Chapter 55
Cyclic Redundancy Check (CRC)

55.1 Chip-specific CRC information

This chip supports one instance of CRC module.

55.2 Introduction

The Cyclic Redundancy Check (CRC) module generates 16/32-bit CRC code for error detection.

The CRC module provides a programmable polynomial and other parameters required to implement a 16-bit or 32-bit CRC standard.

The 16/32-bit code is calculated for 32 bits of data at a time.

55.2.1 Features

Following are the features of the CRC module.

- Hardware CRC generator circuit using a 16-bit or 32-bit programmable shift register
- Programmable initial seed value and polynomial
- Option to transpose input data or output data (the CRC result) bitwise or bytewise. This option is required for certain CRC standards. A bytewise transpose operation is not possible when accessing the CRC data register via 8-bit accesses. In this case, the user's software must perform the bytewise transpose function.
- Option for inversion of final CRC result
- 32-bit CPU register programming interface

55.2.2 Block diagram

The following is a block diagram of the CRC.

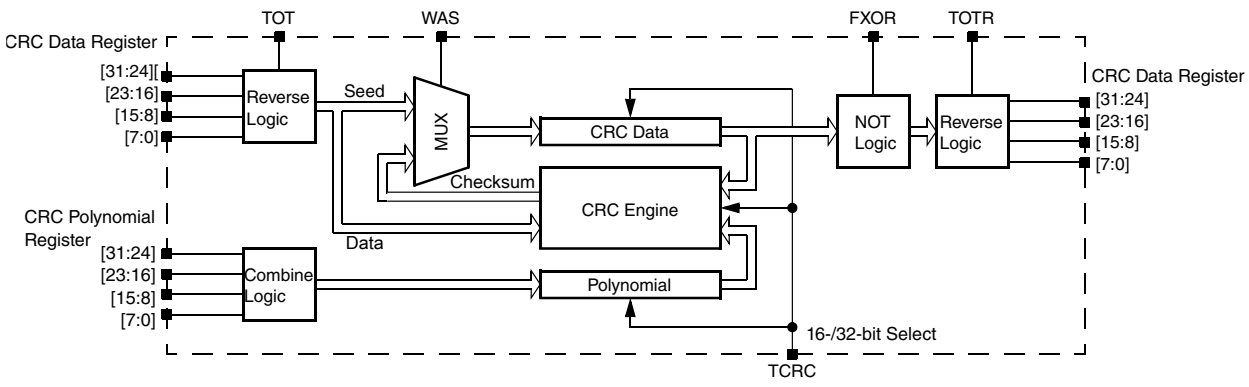


Figure 204. Programmable cyclic redundancy check (CRC) block diagram

55.2.3 Modes of operation

Following sections provide a brief description of various modes which affect the functionality of the CRC module.

55.2.3.1 Run mode

This is the basic mode of operation.

55.2.3.2 Low-power modes

Any CRC calculation in progress stops when the device enters a low-power mode that disables the module clock. It resumes after the clock is enabled or via the system reset for exiting the low-power mode. Clock gating for this module is dependent on the device.

55.3 Functional description

55.3.1 CRC initialization/reinitialization

To enable the CRC calculation, the user must program CRC_CTRL[WAS], CRC_GPOLY, necessary parameters for transposition and CRC result inversion in the applicable registers. Asserting CRC_CTRL[WAS] enables the programming of the seed value into the CRC_DATA register.

After a completed CRC calculation, the module can be reinitialized for a new CRC computation by reasserting CRC_CTRL[WAS] and programming a new, or previously used, seed value. All other parameters must be set before programming the seed value and subsequent data values.

55.3.2 CRC calculations

In 16-bit and 32-bit CRC modes, data values can be programmed 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous. Noncontiguous bytes can lead to an incorrect CRC computation.

55.3.2.1 16-bit CRC

The steps to compute a 16-bit CRC are listed as follows.

1. Clear CRC_CTRL[TCRC] to enable 16-bit CRC mode.
2. Program the transpose and complement options in the CTRL register as required for the CRC calculation. See Transpose feature and CRC result complement for details.
3. Write a 16-bit polynomial to the CRC_GPOLY[LOW] field. The CRC_GPOLY[HIGH] field is not usable in 16-bit CRC mode.
4. Set CRC_CTRL[WAS] to program the seed value.
5. Write a 16-bit seed to CRC_DATA[LU:LL]. CRC_DATA[HU:HL] are not used.
6. Clear CRC_CTRL[WAS] to start writing data values.
7. Write data values into CRC_DATA[HU:HL:LU:LL]. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC_DATA[LU:LL].
8. When all values have been written, read the final CRC result from CRC_DATA[LU:LL].

Transpose and complement operations are performed on-the-fly while reading or writing values. See Transpose feature and CRC result complement for details.

55.3.2.2 32-bit CRC

The steps to compute a 32-bit CRC are listed as follows.

1. Set CRC_CTRL[TCRC] to enable 32-bit CRC mode.
2. Program the transpose and complement options in the CTRL register as required for the CRC calculation. See Transpose feature and CRC result complement for details.
3. Write a 32-bit polynomial to CRC_GPOLY[HIGH:LOW].
4. Set CRC_CTRL[WAS] to program the seed value.
5. Write a 32-bit seed to CRC_DATA[HU:HL:LU:LL].
6. Clear CRC_CTRL[WAS] to start writing data values.

7. Write data values into CRC_DATA[HU:HL:LU:LL]. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC_DATA[HU:HL:LU:LL].
8. When all values have been written, read the final CRC result from CRC_DATA[HU:HL:LU:LL]. The CRC is calculated bytewise, and two clocks are required to complete one CRC calculation.

Transpose and complement operations are performed on-the-fly while reading or writing values. See Transpose feature and CRC result complement for details.

55.3.3 Transpose feature

By default, the transpose feature is not enabled. However, some CRC standards require the input data and/or the final checksum to be transposed. The user software has the option to configure each transpose operation separately, as desired by the CRC standard. The data is transposed on-the-fly while being read or written.

Some protocols use little-endian format for the data stream to calculate a CRC. In this case, the transpose feature usefully flips the bits. This transpose option is one of the types supported by the CRC module.

55.3.3.1 Types of transpose

The CRC module provides several types of transpose functions to flip the bits and/or bytes, for both writing input data and reading the CRC result, separately using the CTRL[TOT] or CTRL[TOTR] fields, according to the CRC calculation being used.

The following types of transpose functions are available for writing to and reading from the CRC Data register.

1. CTRL[TOT] or CTRL[TOTR] is 0b00.

   No transposition occurs.

2. CTRL[TOT] or CTRL[TOTR] is 0b01

   Bits in a byte are transposed, while bytes are not transposed.

   reg[31:0] becomes {reg[24:31], reg[16:23], reg[8:15], reg[0:7]}. See the following figure.

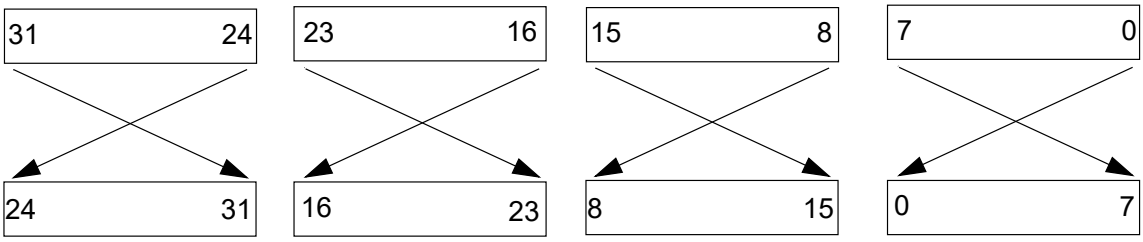


Figure 205. Transpose type 0b01

3. CTRL[TOT] or CTRL[TOTR] is 0b10.

   Both bits in bytes and bytes are transposed.

   reg[31:0] becomes = {reg[0:7], reg[8:15],reg[16:23], reg[24:31]}. See the following figure.

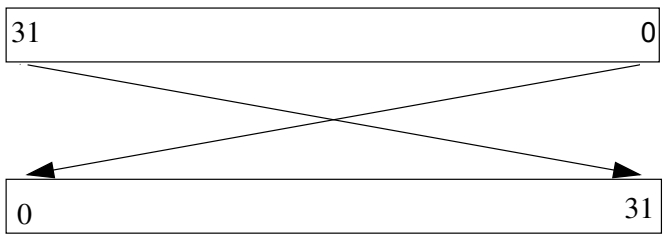


Figure 206. Transpose type 0b10

4. CTRL[TOT] or CTRL[TOTR] is 0b11.

Bytes are transposed, but bits are not transposed.

reg[31:0] becomes {reg[7:0], reg[15:8], reg[23:16], reg[31:24]}. See the following figure.

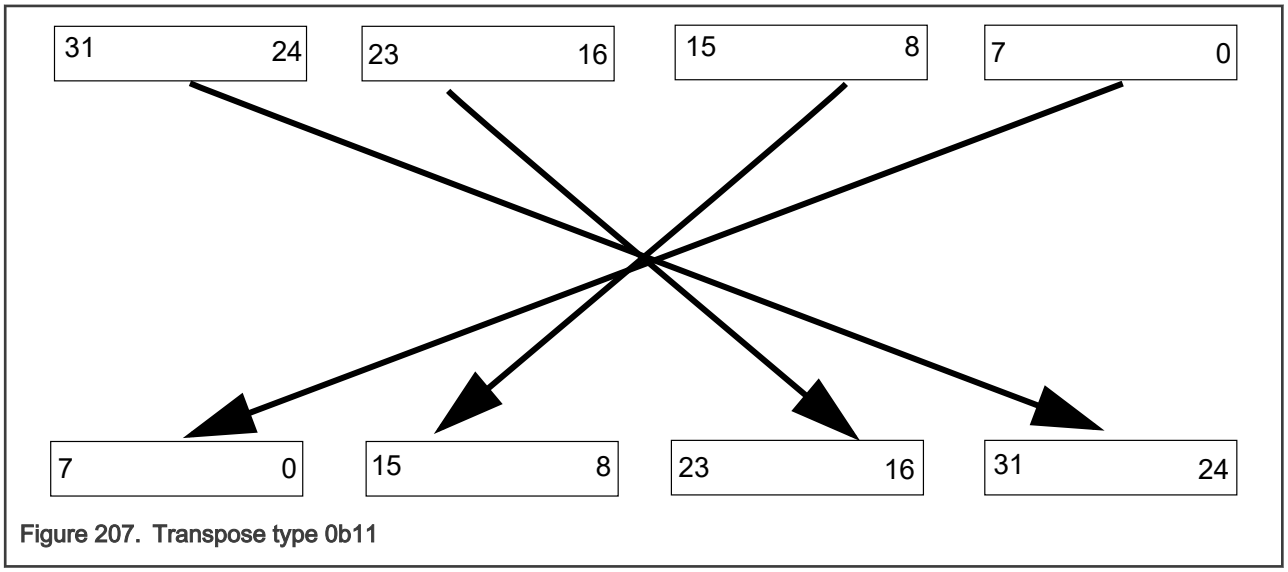


Figure 207. Transpose type 0b11

NOTE

For 8-bit and 16-bit write accesses to the CRC data register, the data is transposed with zeros on the unused byte or bytes (taking 32 bits as a whole), but the CRC is calculated on the valid byte(s) only. When reading the CRC data register for a 16-bit CRC result and using transpose options 10 and 11, the resulting value after transposition resides in the CRC[HU:HL] fields. The user software must account for this situation when reading the 16-bit CRC result, so reading 32 bits is preferred.

55.3.4 CRC result complement

When CTRL[FXOR] is set, the checksum is complemented. The CRC result complement function outputs the complement of the checksum value stored in the CRC Data (DATA) Register every time the CRC Data (DATA) register is read. When CTRL[FXOR] is cleared, reading the CRC Data (DATA) Register accesses the raw checksum value.

55.4 Use cases

NOTE

The following tables apply to little-endian format.

55.4.1 CRC Control Register programming

The following table shows the CRC Control Register programming for 16-bit CRC.

Table 256. Control Register (CTRL) programming for 16-bit CRC

Algorithm	Poly	Seed	Ref In	Ref Out	XOR Out	CTRL[TOT]	CTRL[TOTR]	CTRL[FXOR]
CRC-16/ CCITT_FALSE	0x1021	0xFFFF	0	0	0x0000	0x0	0x0	0x0
CRC-16 / ARC	0x8005	0x0000	1	1	0x0000	0x1	0x2	0x0
CRC16_AUG_CCITT	0x1021	0x1D0F	0	0	0x0000	0x0	0x0	0x0

Table continues on the next page...

Table 256. Control Register (CTRL) programming for 16-bit CRC (continued)

Algorithm	Poly	Seed	Ref In	Ref Out	XOR Out	CTRL[TOT]	CTRL[TOTR]	CTRL[FXOR]
CRC16_BUYPASS	0x8005	0x0000	0	0	0x0000	0x0	0x0	0x0
CRC16_CCITT_ZERO	0x1021	0x0000	0	0	0x0000	0x0	0x0	0x0
CRC16_CDMA2000	0xC867	0xFFFF	0	0	0x0000	0x0	0x0	0x0
CRC16_DDS_110	0x8005	0x800D	0	0	0x0000	0x0	0x0	0x0
CRC16_DECT_R	0x589	0x0000	0	0	0xFFFF	0x1	0x2	0x1
CRC16_DECT_X	0x589	0x0000	0	0	0x0000	0x0	0x0	0x0
CRC16_DNP	0x3D65	0x0000	1	1	0xFFFF	0x1	0x2	0x1
CRC16_EN_13757	0x3D65	0x0000	0	0	0xFFFF	0x1	0x2	0x1
CRC16_GENIBUS	0x1021	0xFFFF	0	0	0xFFFF	0x1	0x2	0x1
CRC16_MAXIM	0x8005	0x0000	1	1	0xFFFF	0x1	0x2	0x1
CRC16_MCRF4XX	0x1021	0xFFFF	1	1	0x0000	0x1	0x2	0x0
CRC16_RIELLO	0x1021	0xB2AA	1	1	0x0000	0x1	0x2	0x0
CRC16_T10_DIF	0x8BB7	0x0000	0	0	0x0000	0x0	0x0	0x0
CRC16_TELEDISK	0xA097	0x0000	0	0	0x0000	0x0	0x0	0x0
CRC16_TMS37157	0x1021	0x89EC	1	1	0x0000	0x1	0x2	0x0
CRC16_USB	0x8005	0xFFFF	1	1	0xFFFF	0x1	0x2	0x1
CRC16_A	0x1021	0xC6C6	1	1	0x0000	0x1	0x2	0x0
CRC16_KERMIT	0x1021	0x0000	1	1	0x0000	0x1	0x2	0x0
CRC16_MODBUS	0x8005	0xFFFF	1	1	0x0000	0x1	0x2	0x0
CRC16_X_25	0x1021	0xFFFF	1	1	0xFFFF	0x1	0x2	0x1
CRC16_XMODEM	0x1021	0x0000	0	0	0x0000	0x0	0x0	0x0

The following table shows the CRC Control Register programming for 32-bit CRC.

Table 257. Control Register (CTRL) programming for 32-bit CRC

Algorithm	Poly	Seed	Ref In	Ref Out	Xor Out	CTRL[TOT]	CTRL[TOTR]	CTRL[FXOR]
CRC - 32	0x04C11DB7	0xFFFFFFFF	1	1	0xFFFF_FFFF	0x1	0x2	0x1
CRC-32/BZIP2	0x04C11DB7	0xFFFFFFFF	0	0	0xFFFF_FFFF	0x0	0x0	0x1

Table continues on the next page...

Table 257. Control Register (CTRL) programming for 32-bit CRC (continued)

Algorithm	Poly	Seed	Ref In	Ref Out	Xor Out	CTRL[TOT]	CTRL[TOTR]	CTRL[FXOR]
CRC-32C	0x1EDC6F41	0xFFFFFFFF	1	1	0xFFFF_FFFF	0x1	0x2	0x1
CRC-32D	0xA833982B	0xFFFFFFFF	1	1	0xFFFF_FFFF	0x1	0x2	0x1
CRC-32/MPEG-2	0x04C11DB7	0xFFFFFFFF	0	0	0x0000_0000	0x0	0x0	0x0
CRC-32/POSIX	0x04C11DB7	0x00000000	0	0	0xFFFF_FFFF	0x0	0x0	0x1
CRC-32Q	0x814141AB	0x00000000	0	0	0x0000_0000	0x0	0x0	0x0
CRC-32/JAMCRC	0x04C11DB7	0xFFFFFFFF	1	1	0x0000_0000	0x1	0x2	0x0
CRC-32/XFER	0x000000AF	0x00000000	0	0	0x0000_0000	0x0	0x0	0x0

55.4.2 Expected read data fields

The following table shows the expected read data fields for 16-bit CRC.

Table 258. Expected read data fields for 16-bit CRC

Algorithm	CRC DATA Register (DATA)
CRC-16/CCITT_FALSE	[31 : 16] = Unknown [15 : 0] = Valid Data
CRC-16 / ARC	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_AUG_CCITT	[31 : 16] = Unknown [15 : 0] = Valid Data
CRC16_BUYPASS	[31 : 16] = Unknown [15 : 0] = Valid Data
CRC16_CCITT_ZERO	[31 : 16] = Unknown [15 : 0] = Valid Data
CRC16_CDMA2000	[31 : 16] = Unknown [15 : 0] = Valid Data
CRC16_DDS_110	[31 : 16] = Unknown [15 : 0] = Valid Data
CRC16_DECT_R	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_DECT_X	[31 : 16] = Unknown [15 : 0] = Valid Data
CRC16_DNP	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_EN_13757	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_GENIBUS	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_MAXIM	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_MCRF4XX	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_RIELLO	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_T10_DIF	[31 : 16] = Unknown [15 : 0] = Valid Data

Table continues on the next page...

Table 258. Expected read data fields for 16-bit CRC (continued)

Algorithm	CRC DATA Register (DATA)
CRC16_TELEDISK	[31 : 16] = Unknown [15 : 0] = Valid Data
CRC16_TMS37157	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_USB	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_A	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_KERMIT	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_MODBUS	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_X_25	[31 : 16] = Valid Data [15 : 0] = Unknown
CRC16_XMODEM	[31 : 16] = Unknown [15 : 0] = Valid Data

The following table shows the expected read data fields for 32-bit CRC.

Table 259. Expected read data fields for 32-bit CRC

Algorithm	CRC DATA Register (DATA)
CRC - 32	[31 : 0] = Valid Data
CRC-32/BZIP2	[31 : 0] = Valid Data
CRC-32C	[31 : 0] = Valid Data
CRC-32D	[31 : 0] = Valid Data
CRC-32/MPEG-2	[31 : 0] = Valid Data
CRC-32/POSIX	[31 : 0] = Valid Data
CRC-32Q	[31 : 0] = Valid Data
CRC-32/JAMCRC	[31 : 0] = Valid Data
CRC-32/XFER	[31 : 0] = Valid Data

55.5 Memory map and register descriptions

NOTE

Write access to the register addresses not mapped to the peripheral but included in the address space of the peripheral may result in unpredictable functionality. The CRC module should be reconfigured if a transfer error occurs.

55.5.1 CRC register descriptions

55.5.1.1 CRC memory map

CRC base address: 4038_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	CRC DATA register (DATA)	32	RW	FFFF_FFFFh
4h	CRC Polynomial register (GPOLY)	32	RW	0000_1021h
8h	CRC Control register (CTRL)	32	RW	0000_0000h

55.5.1.2 CRC DATA register (DATA)

Offset

Register	Offset
DATA	0h

Function

The CRC DATA register contains the value of the seed, data, and checksum. When CTRL[WAS] is set, any write to the data register is regarded as the seed value. When CTRL[WAS] is cleared, any write to the data register is regarded as data for general CRC computation.

In 16-bit CRC mode, the HU and HL fields are not used for programming the seed value, and reads of these fields return an indeterminate value. In 32-bit CRC mode, all fields are used for programming the seed value.

When programming data values, the values can be written 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous; with MSB of data value written first.

After all data values are written, the CRC result can be read from this data register. In 16-bit CRC mode, the CRC result is available in the LU and LL fields. In 32-bit CRC mode, all fields contain the result. Reads of this register at any time return the intermediate CRC value, provided the CRC module is configured.

Diagram

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

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

Fields

Field	Function
31-24 HU	CRC High Upper Byte In 16-bit CRC mode (CTRL[TCRC] is 0), this field is not used for programming a seed value. In 32-bit CRC mode (CTRL[TCRC] is 1), values written to this field are part of the seed value when CTRL[WAS] is 1. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation in both 16-bit and 32-bit CRC modes.
23-16 HL	CRC High Lower Byte In 16-bit CRC mode (CTRL[TCRC] is 0), this field is not used for programming a seed value. In 32-bit CRC mode (CTRL[TCRC] is 1), values written to this field are part of the seed value when CTRL[WAS] is 1. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation in both 16-bit and 32-bit CRC modes.
15-8 LU	CRC Low Upper Byte When CTRL[WAS] is 1, values written to this field are part of the seed value. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation.
7-0 LL	CRC Low Lower Byte When CTRL[WAS] is 1, values written to this field are part of the seed value. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation.

55.5.1.3 CRC Polynomial register (GPOLY)

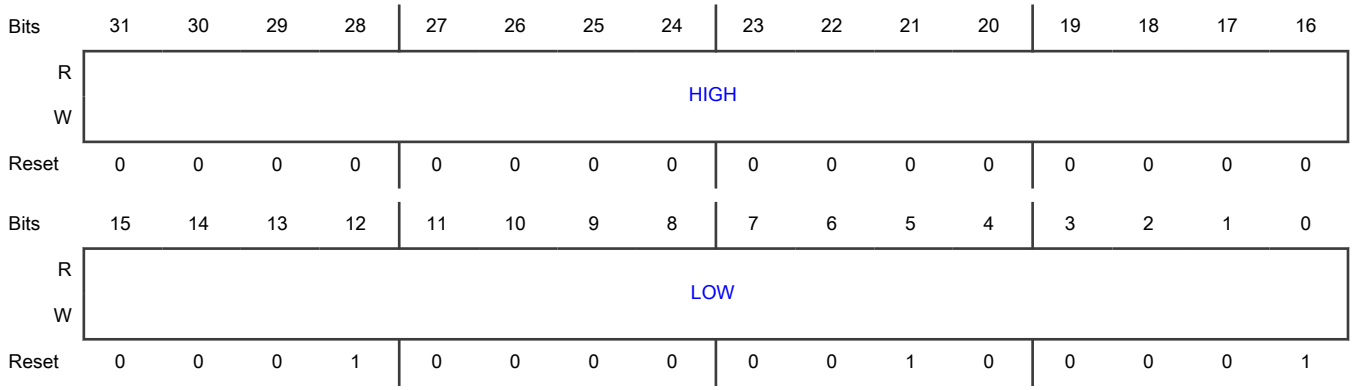
Offset

Register	Offset
GPOLY	4h

Function

This register contains the value of the polynomial for the CRC calculation. The HIGH field contains the upper 16 bits of the CRC polynomial, which are used only in 32-bit CRC mode. Writes to the HIGH field are ignored in 16-bit CRC mode. The LOW field contains the lower 16 bits of the CRC polynomial, which are used in both 16- and 32-bit CRC modes.

Diagram



Fields

Field	Function
31-16 HIGH	High Polynominal Half-word Writable and readable in 32-bit CRC mode (CTRL[TCRC] is 1). This field is not writable in 16-bit CRC mode (CTRL[TCRC] is 0).
15-0 LOW	Low Polynominal Half-word Writable and readable in both 32-bit and 16-bit CRC modes.

55.5.1.4 CRC Control register (CTRL)

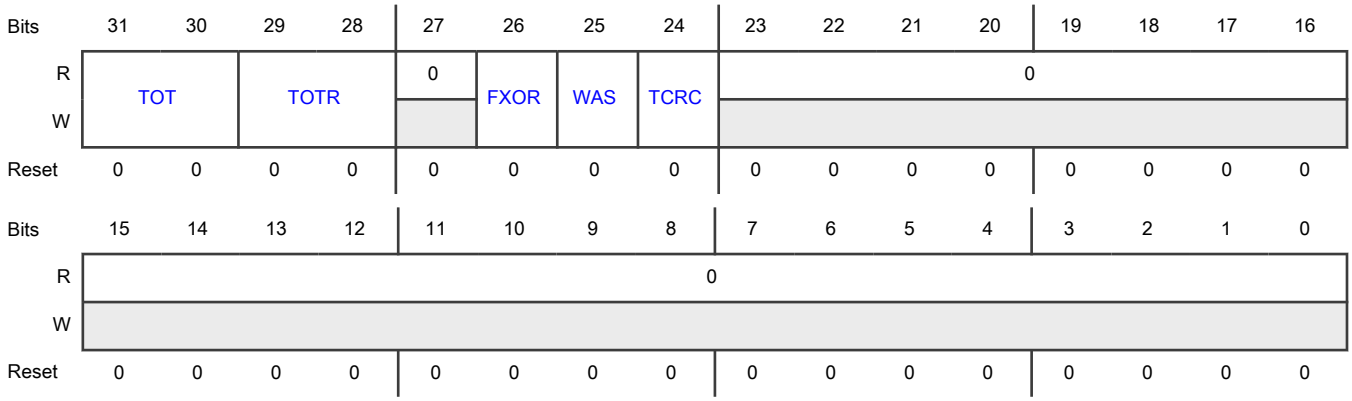
Offset

Register	Offset
CTRL	8h

Function

This register controls the configuration and working of the CRC module. Appropriate bits must be set before starting a new CRC calculation. A new CRC calculation is initialized by asserting CTRL[WAS] and then writing the seed into the CRC DATA register.

Diagram



Fields

Field	Function
31-30 TOT	Type Of Transpose For Writes Defines the transpose configuration of the data written to the CRC DATA register. See Transpose feature for the available transpose options. 00b - No transposition.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Bits in bytes are transposed; bytes are not transposed. 10b - Both bits in bytes and bytes are transposed. 11b - Only bytes are transposed; no bits in a byte are transposed.
29-28 TOTR	Type Of Transpose For Read Identifies the transpose configuration of the value read from the CRC DATA register. See Transpose feature for the available transpose options. 00b - No transposition. 01b - Bits in bytes are transposed; bytes are not transposed. 10b - Both bits in bytes and bytes are transposed. 11b - Only bytes are transposed; no bits in a byte are transposed.
27 —	Reserved
26 FXOR	Complement Read Of CRC DATA register Some CRC protocols require the final checksum to be XORed with 0xFFFFFFFF or 0xFFFF. Asserting this bit enables on-the-fly complementing of read data. 0b - No XOR on reading. 1b - Invert or complement the read value of the CRC DATA register.
25 WAS	Write CRC DATA register As Seed When asserted, a value written to the CRC DATA register is considered a seed value. When deasserted, a value written to the CRC DATA register is taken as data for CRC computation. 0b - Writes to the CRC DATA register are data values. 1b - Writes to the CRC DATA register are seed values.
24 TCRC	TCRC Width of CRC protocol. 0b - 16-bit CRC protocol. 1b - 32-bit CRC protocol.
23-0 —	Reserved

55.6 Glossary

Transpose	Flipping input or output data bits (or bytes)
Seed value	Initial value of CRC calculation

Chapter 56
Power Conversion and Motor Control (PCMC)

56.1 Introduction

The PCMC subsystem is a group of modules that can be interconnected to perform a variety of real-time tasks, such as:

- Motor control
- Power conversion
- Advanced PWM generation
- Lighting control

The modules that make up PCMC are:

- ADC
- BCTU
- eMIOS
- LCU
- LPCMP
- TRGMUX

See Feature comparison for the number of instances of each module in your chip.

56.2 Block diagram

The figure and table below show a typical configuration.

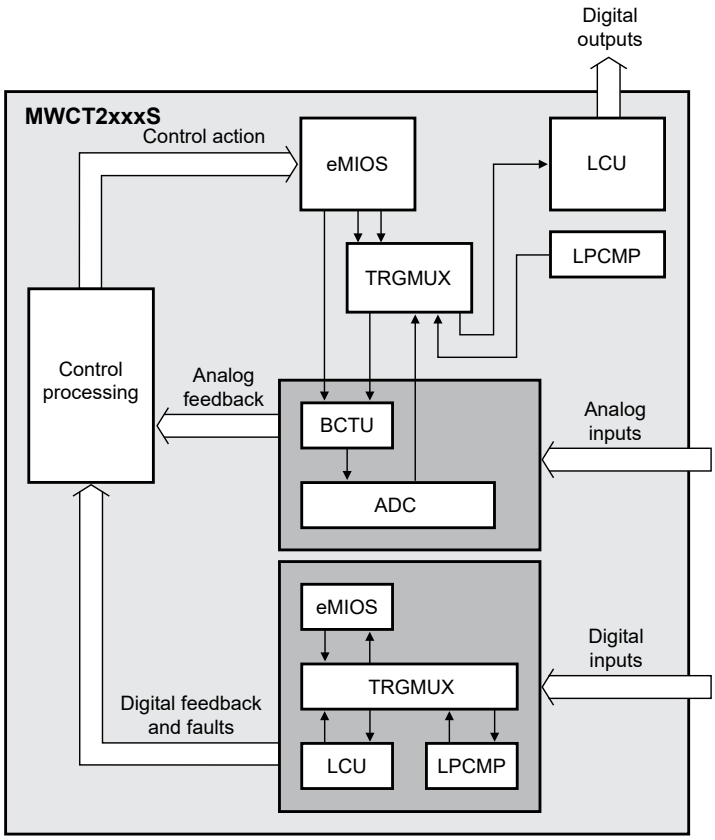


Figure 208. Block diagram

Table 260. Block diagram components

Component	Description
ADC	Measures external analog signals
BCTU	Performs advanced ADC triggering
Control processing	Arm Cortex-M7 core: • Processes digital and analog inputs to generate control outputs • Writes control outputs to peripherals
eMIOS	• Generates high-resolution PWM outputs • Measures time-based digital inputs
LCU	Performs programmable logic functions and fault control
LPCMP	Compares analog values to detect faults
TRGMUX	• Routes signals between internal peripherals • Routes external signals to internal peripherals

56.3 Functional overview

56.3.1 Analog signal measurement

ADC measures the voltage of external analog signals and converts that voltage to a digital value. Each ADC instance operates independently of other instances. Software or hardware can trigger conversions. Together with BCTU and eMIOS, ADC can preempt normal measurements by injecting high-priority measurements when needed.

See Analog-to-Digital Converter.

56.3.2 Advanced ADC triggering

BCTU works with ADC to initiate conversions triggered by outputs from other modules in the PCMC subsystem, such as PWM signals from eMIOS or LCU sequential logic outputs.

With BCTU, you can group measurements into configuration lists containing up to 32 items. Each item contains information such as:

- ADC instance to use for the conversion
- The ADC channel
- The trigger event for initiating the next conversion
- The conversion list address

BCTU can store conversion results by routing them through DMA or putting them in FIFO queues. The FIFO queues enable conversions to occur seamlessly by storing conversion results without software intervention.

BCTU has 72 trigger inputs, 69 of which come from the eMIOS channel outputs. The three remaining trigger inputs come from other on-chip modules via TRGMUX. You can also trigger conversions by software through the BCTU register interface.

BCTU can issue a single conversion or a list of conversions based on a single trigger occurrence. BCTU signals the eMIOS timers when ADC receives a single command or a list of commands. ADC sends conversion data back to BCTU, which loads it in the ADC*n* Result Data (ADC*n*DR) register for the CPU to read.

BCTU-to-ADC interface illustrates the connections between BCTU and an ADC.

See also:

- Body cross-triggering unit (BCTU)
- PCDR*n*, ICDR*n*, and ECDR*n* register descriptions in ADC register descriptions

56.3.3 BCTU-to-ADC interface

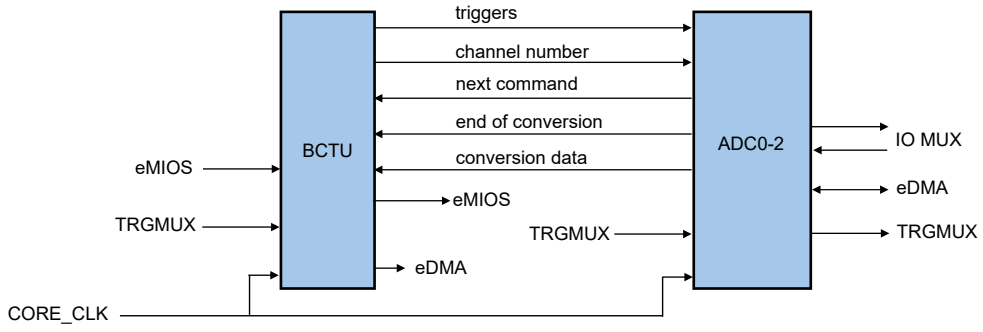


Figure 209. BCTU-to-ADC interface

56.3.4 Generate PWM outputs and measure digital inputs

eMIOS generates high-resolution PWM outputs. Each eMIOS instance has 23 dedicated channel outputs. You can configure each channel to operate in a different mode. Depending on the mode, eMIOS channels can:

- Measure the period of an input signal
- Measure the width of an input pulse
- Function as general-purpose I/O
- Count pulses or edges
- Capture input values
- Compare outputs

For example, eMIOS can generate periodic ADC triggers. Using TRGMUX, you can route eMIOS triggers to other on-chip peripherals such as LCU, another eMIOS instance, external chip I/O, and so on.

eMIOS connections illustrates the input and output connections for eMIOS.

See also:

- Enhanced Modular IO Subsystem (eMIOS)
- Body cross-triggering unit (BCTU)
- Trigger Mux (TRGMUX)

56.3.5 eMIOS connections

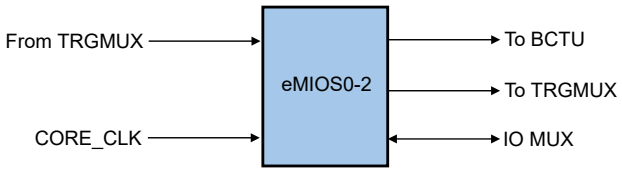


Figure 210. eMIOS connections

56.3.6 Create combinatorial and sequential logic functions

LCU enables you to implement hardware-based combinatorial and sequential logic on the chip. This capability reduces cost and decreases application size.

Each LCU instance has 12 dedicated inputs and 12 outputs. You can use LCU to implement a variety of combinatorial and sequential logic functions with no CPU intervention, including a simple OR gate, flip flops, and motor control logic. As detailed in Fault detection, LCU also plays a significant role in fault management.

LCU connections shows the input and output connections for LCU.

See also:

- Logic Control Unit (LCU)
- LCU use case examples
- Trigger MUX (TRGMUX)
- TRGMUX connectivity file attached to this document

56.3.7 LCU connections

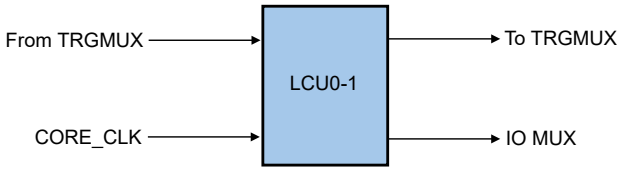


Figure 211. LCU connections

56.3.8 Fault detection

The PCMC subsystem uses LPCMP and LCU to implement fault detection.

LPCMP can detect fault conditions (such as short circuits, over-voltages, and excessive temperature) by comparing an external input voltage to a programmable reference voltage. It produces a digital output that indicates whether the input voltage is higher or lower than the reference voltage.

The chip can detect faults via external digital or analog signals. TRGMUX routes digital signals from SIUL to a peripheral to trigger the appropriate behavior.

LCU has four force inputs that you can use to manage fault assertion via its own internal logic mechanism.

The typical reaction to the presence of fault is to put the system in a safe state by:

- Deasserting output signals such as PWM outputs controlling a power stage.
- Disconnecting an entire circuit by switching off a relay.

See also Low Power Comparator (LPCMP).

56.3.9 Programmable module interconnections

TRGMUX enables you to interconnect on-chip peripherals to create application-specific configurations. TRGMUX has 128 inputs and 111 outputs, which you can interconnect in hundreds of combinations through software during runtime.

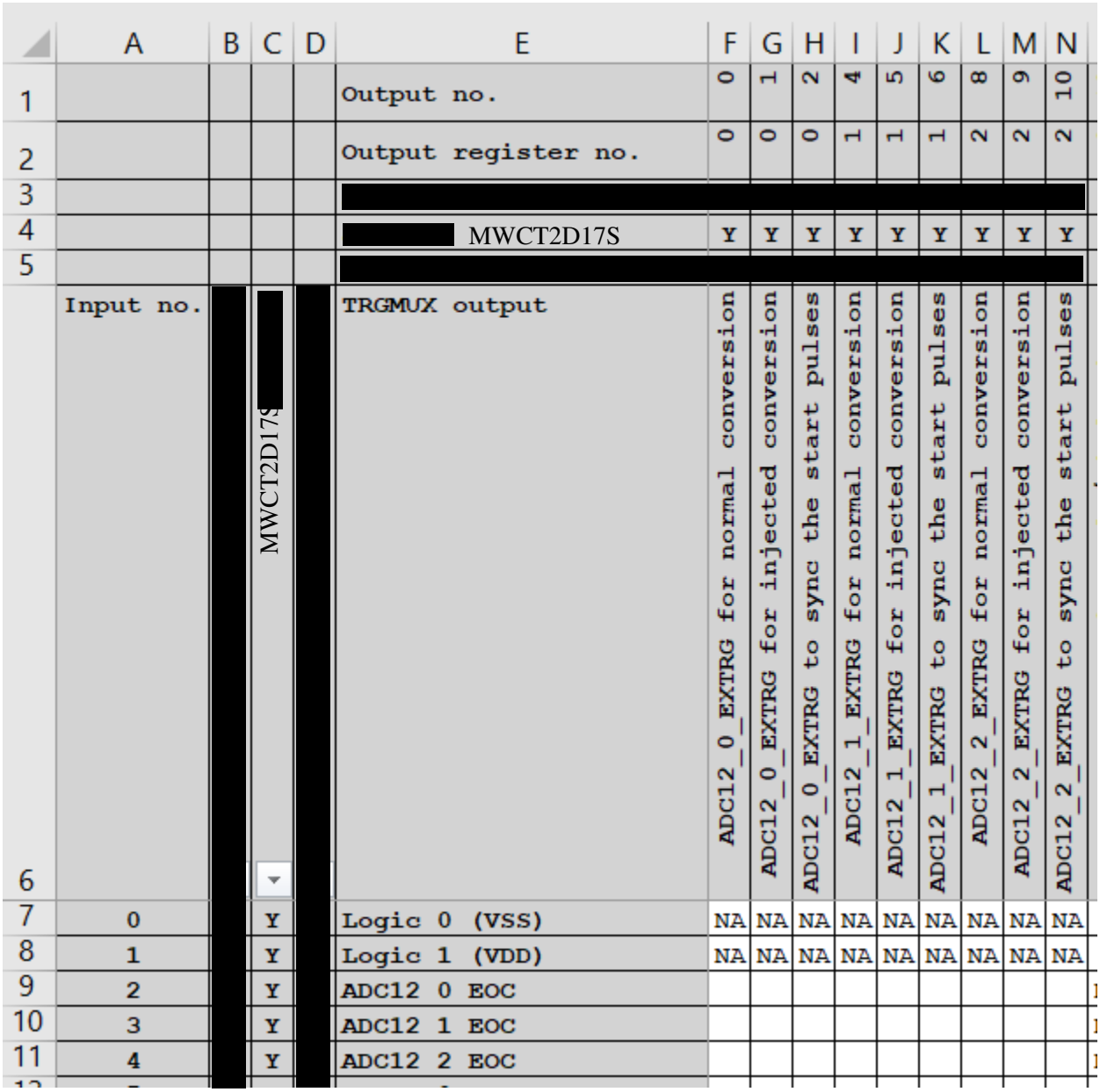
You can also route on-chip peripheral outputs to chip I/O pins to enable signal debugging. Similarly, you can route external signals to on-chip peripheral inputs.

NOTE

There is a two-cycle AIPS_SLOW_CLK delay through TRGMUX.

For details of PCMC interconnections, see PCMC connections diagram.

The attached TRGMUX connectivity file, a portion of which appears below, shows the interconnections possible between module instances. Column B in the spreadsheet lists TRGMUX inputs. Row 3 lists TRGMUX outputs. A connection between an input and an output is possible if the intersecting cell of the input row and the output column is blank. "NA" indicates that a connection is not possible.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1					Output no.	0	1	2	4	5	6	8	9	10
2					Output register no.	0	0	0	1	1	1	2	2	2
3					[redacted]									
4					[redacted] MWCT2D17S	Y	Y	Y	Y	Y	Y	Y	Y	Y
5					[redacted]									
6	Input no.	[redacted]	[redacted] MWCT2D17S	[redacted]	TRGMUX output	ADC12_0_EXTRG for normal conversion	ADC12_0_EXTRG for injected conversion	ADC12_0_EXTRG to sync the start pulses	ADC12_1_EXTRG for normal conversion	ADC12_1_EXTRG for injected conversion	ADC12_1_EXTRG to sync the start pulses	ADC12_2_EXTRG for normal conversion	ADC12_2_EXTRG for injected conversion	ADC12_2_EXTRG to sync the start pulses
7	0		Y		Logic 0 (VSS)	NA	NA	NA	NA	NA	NA	NA	NA	NA
8	1		Y		Logic 1 (VDD)	NA	NA	NA	NA	NA	NA	NA	NA	NA
9	2		Y		ADC12 0 EOC									
10	3		Y		ADC12 1 EOC									
11	4		Y		ADC12 2 EOC									

Figure 212. Portion of TRGMUX connectivity

See also:

- Trigger Mux (TRGMUX)
- TRGMUX connectivity file attached to this document

56.3.9.1 PCMC connections diagram

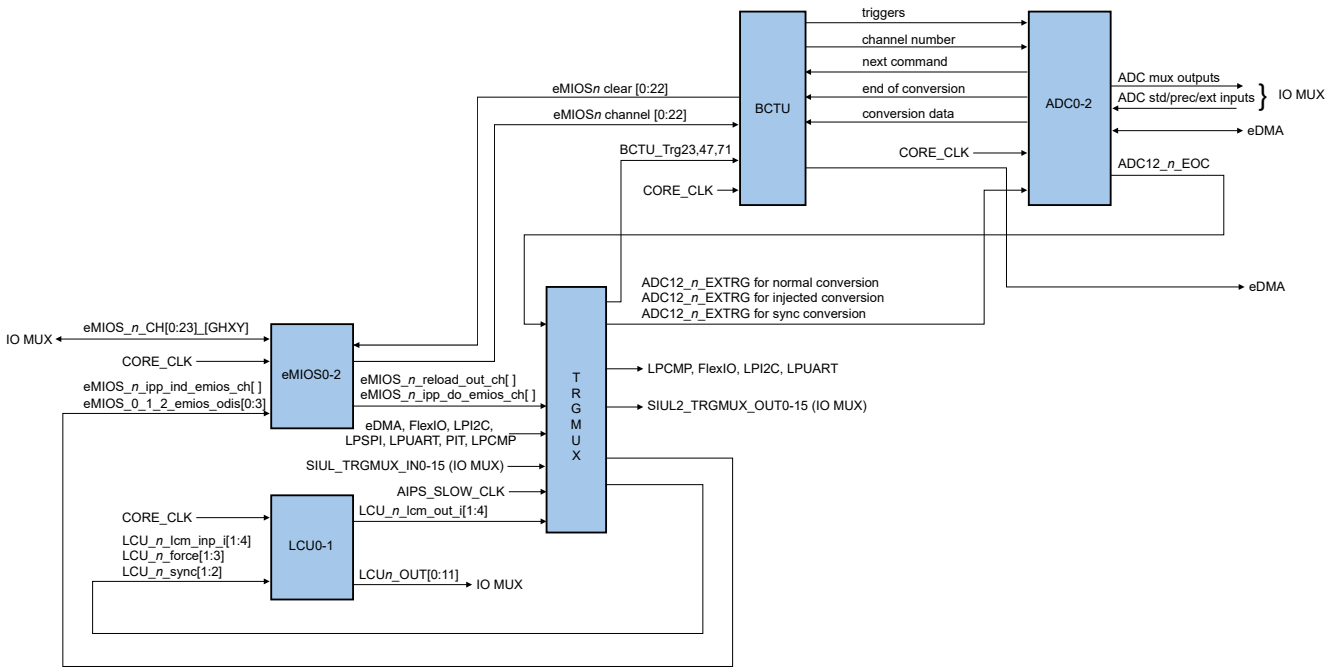


Figure 213. PCMC connections diagram

56.4 Reset, clocking, and other considerations

56.4.1 Reset

ADC, BCTU, eMIOS, LCU, and TRGMUX fully reset during a POR, destructive reset, or functional reset.

See also:

- Module reset status
- Software reset

56.4.2 Clocking

Detailed clocking information is given in the links below. In summary:

- CORE_CLK clocks ADC, BCTU, eMIOS, and LCU.
- AIPS_SLOW_CLK clocks TRGMUX.
- In general, you must set AIPS_SLOW_CLK to one-fourth or one-half the frequency of CORE_CLK.

See also:

- ADC and motor control modules
- Clocking use case examples
- System clock frequency limitations
- Clock frequency

56.4.3 Power modes

All ADC, BCTU, eMIOS, LCU, and TRGMUX instances operate only in Run mode. None of the instances operate in Standby mode.

See also:

- Module reset status
- ADC modes of operation
- LCU modes of operation

56.5 Use case examples

56.5.1 Motor control use case

The following sections show:

- A PMSM control system
- A BLDC control system

Both PMSM and BLDC use a rotating magnetic stator field to turn the stator consisting of permanent magnets. The stator is driven by 3-phase AC voltage. The difference between PMSM and BLDC is in how stator position is calculated.

- PMSM calculates stator position by measuring back EMF from the motor.
- BLDC calculates stator position based on feedback from motor sensors.

56.5.1.1 PMSM motor control use case

56.5.1.1.1 Generic PMSM motor control

The following figure and table illustrate a generic PMSM motor control system. The feedback loops in the system create a motor control loop.

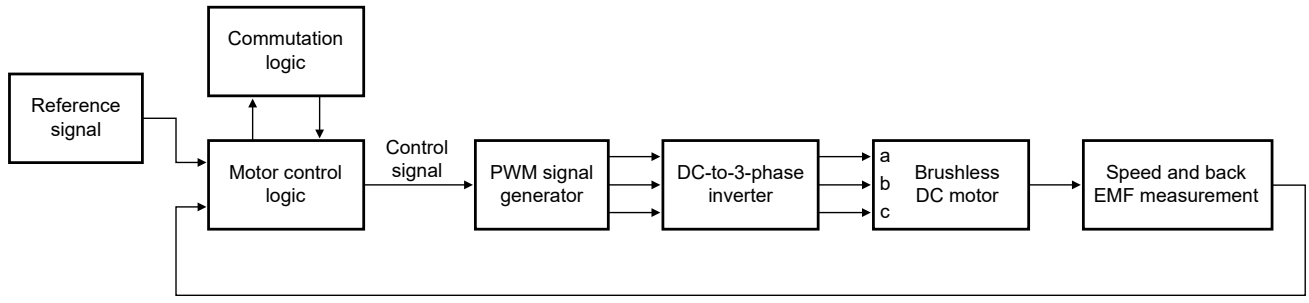


Figure 214. Generic PMSM motor control

Table 261. Block diagram components

Block	Description
Angular position sensor	Generates data that enables commutation logic to calculate rotor position.
Brushless DC motor	Electronically commutated electric motor.
Commutation logic	Compares values from the rotor angular position sensors to entries in a lookup table, and applies voltage to stator winding corresponding to same set of values in lookup table.

Table continues on the next page...

Table 261. Block diagram components (continued)

Block	Description
DC-to-3-phase inverter	Switching network that transmits power to the motor stator windings according to a control signal.
Motor control logic	• Compares speed measurement to required speed and adjusts the PWM signal generator. • Generates the assertions for the PWM signal generator according to commutation logic input.
PWM signal generator	Generates PWM signal voltage source.
Reference signal	Signal that indicates required motor speed.
Speed and back EMF measurement	An ADC is linked to a shunt resistor to perform measurements and detect over-current conditions. If one is found, then hardware generates a fault interrupt.

56.5.1.1.2 PMSM motor control

The PCMC subsystem consists of a group of modules that enable you to implement PMSM motor control loops in a variety of ways. The following figure and table illustrate one possible implementation.

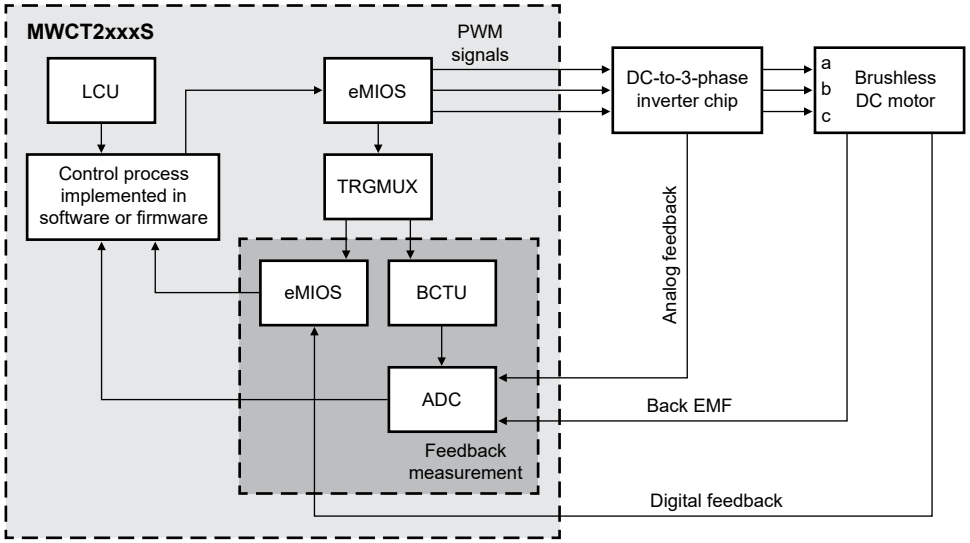


Figure 215. PMSM motor control

Table 262. Block diagram components

Block	Description
ADC	Measures analog inputs (currents and/or voltages) and converts to digital values. Typically, it measures currents and can be configured to generate a fault interrupt when a measured input is outside a pre-defined acceptable range.
BCTU	Triggers conversions by ADC.
Control process	Provides top-level system control functions for FOC or another feasible control algorithm:

Table continues on the next page...

Table 262. Block diagram components (continued)

Block	Description
	• Compares speed measurement to required speed and adjusts the PWM signal generator. • Generates the assertions for the PWM signal generator according to commutation logic input.
DC-to-3-phase inverter	Switching network that transmits power to the motor stator windings according to a control signal.
eMIOS	• Generates PWM input for DC to 3-phase inverter. • Measures digital feedback inputs—BLDC position sensors, for example.
LCU	Implements commutation lookup table. Control Process compares values from the angular position sensors to entries in the lookup table. It then applies voltage to stator winding corresponding to same set of values in lookup table.
TRGMUX	• Enables interconnect to the required set of on-chip peripherals. • Indirectly triggers ADC sampling via BCTU. • Triggers eMIOS to measure digital feedback signals.

56.5.1.2 Sensored BLDC motor control use case

56.5.1.2.1 Generic sensored BLDC motor control

The following figure and table illustrate a generic sensored BLDC motor control system. The feedback loops in the system create a motor control loop.

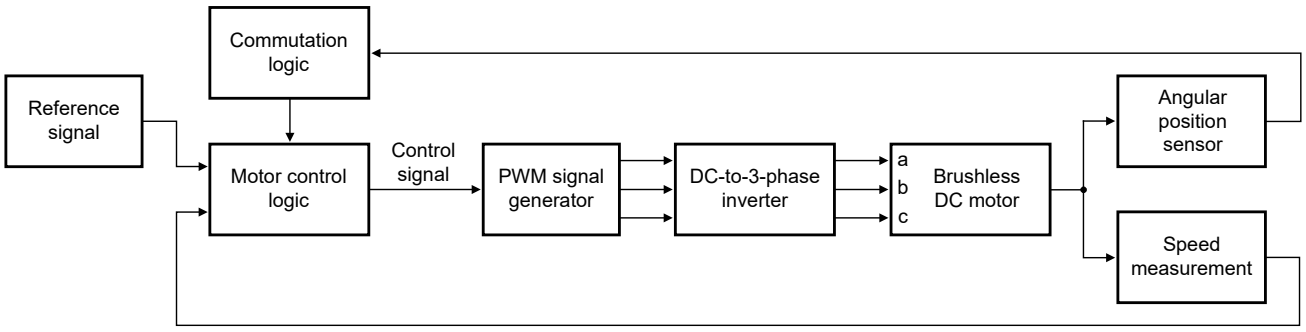


Figure 216. Generic sensored BLDC motor control

Table 263. Block diagram components

Block	Description
Angular position sensor	Hall effect sensors or optical decoders detect rotor position
Commutation logic	Compares values from the angular position sensors to entries in a lookup table and applies voltage to stator winding corresponding to same set of values in lookup table
Reference signal	Signal that indicates required motor speed

Table continues on the next page...

Table 263. Block diagram components (continued)

Block	Description
DC-to-3-phase inverter	Switching network that transmits power to the stator windings according to a control signal
Motor control logic	• Compares speed measurement to required speed and adjusts the PWM signal generator • Generates the assertions for the PWM signal generator according to commutation logic input
Speed measurement	Uses an ADC linked to a shunt resistor to perform measurements and detect over-current conditions, in which case hardware generates an interrupt.

56.5.1.2.2 PCMC sensored field-oriented control (FOC) of 3-phase BLDC motor

FOC is a VFD control method in which the stator currents of a 3-phase AC electric motor (such as BLDC, PMCM, reluctance motor, or other spinning electrical machine) is identified as a vector.

The PCMC subsystem consists of a group of modules that enable you to implement sensored FOC BLDC motor control loops in a variety of ways. The following figure and table show one possible implementation.

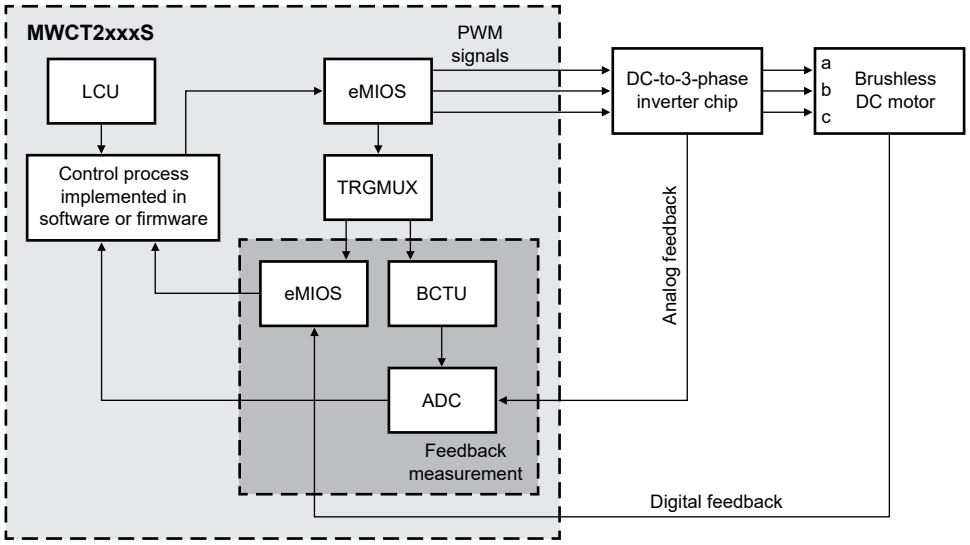


Figure 217. PCMC sensored BLDC motor control

Table 264. Block diagram components

Block	Description
ADC	Measures analog inputs (currents and/or voltages) and converts to digital values. Typically, it measures currents, and can be configured to generate a fault interrupt when a measured input is outside a pre-defined acceptable range.
BCTU	Triggers conversions by ADC.
Control process	Provides top-level system control functions for FOC control:

Table continues on the next page...

Table 264. Block diagram components (continued)

Block	Description
	• Compares speed measurement to required speed and adjusts the PWM signal generator. • Generates the assertions for the PWM signal generator according to commutation logic input.
DC-to-3-phase inverter	Switching network that transmits power to the motor stator windings according to a control signal.
eMIOS	• Generates PWM input for DC to 3-phase inverter. • Measures digital feedback inputs—BLDC position sensors, for example.
LCU	Implements commutation lookup table. Control process compares values from the angular position sensors to entries in the lookup table. It then applies voltage to stator winding corresponding to same set of values in lookup table.
TRGMUX	• Enables interconnect of the required set on on-chip peripherals. • Indirectly triggers ADC sampling via BCTU. • Triggers eMIOS to measure digital feedback signals.

56.6 Design considerations

56.6.1 ADC design considerations

The following topics discuss factors that affect ADC function in applications.

56.6.1.1 ADC measurement accuracy

The FOC motor control algorithm requires a high degree of confidence in the reliability of conversion data. Here are some best practices that contribute to ADC precision.

- Use precision channels (see ADC precision channels).
- Minimize input noise (see ADC input noise).
- Use BCTU triggering (see ADC triggering).

56.6.1.1.1 ADC precision channels

See the chip-specific ADC configuration information for the number of precision channels for each ADC instance.

See the channel mapping table for the channel numbers for the precision channels.

56.6.1.1.2 ADC input noise

Regardless of the application, ADCs are inherently sensitive to input noise. The following best practices help minimize input noise.

- Avoid simultaneous conversions.
- Consider using a low-pass filter on input pins.

All variants of this chip include multiple, independent ADC instances, so it is easily possible to have simultaneous sampling by different ADC instances. This should be avoided, though, because ADC sampling can cause noise on adjacent device ADC inputs. Avoidance of simultaneous sampling should be ensured by software design. This is important for implementations of the FOC algorithm.

Additionally, an RC low-pass filter on ADC input pins can help reduce noise and enhance accuracy.

56.6.1.1.3 ADC triggering

Some applications, including FOC motor control, must use BCTU for precise triggering of ADC measurement. This is because the FOC algorithm requires sampling at specific times to implement the “observer/predictor”. BCTU makes it possible to perform precisely-timed ADC measurement.

For less exigent algorithms or non-closed-loop controlled motors, BCTU triggering and precision ADC channels are not necessarily required.

56.6.1.2 Example ADC configurations

Following are some ADC configurations that might be used for FOC motor control.

Table 265. Example ADC configurations

Description	Settings
Single-shot mode	• Enable Overwrite (OWREN=1) • Disable Write Left-Aligned (WLSIDE=0) • Select single shot (MODE=0) • Disable External Trigger (TRGEN=0) • Disable Injection Trigger (JTRGEN=0) • Disable BCTU triggering (BCTUEN=0) • Disable averaging (AVGEN=0) • Select conversion clock = 1/2 module clock (ADCCLKSEL=01)
Normal conversion scan mode	• Enable Overwrite (OWREN=1) • Disable Write Left-Aligned (WLSIDE=0) • Select looped conversions (MODE=1) • Disable External Trigger (TRGEN=0) • Disable Injection Trigger (JTRGEN=0) • Disable BCTU triggering (BCTUEN=0) • Disable averaging (AVGEN=0) • Select conversion clock = 1/2 module clock (ADCCLKSEL=01)
BCTU control mode	• Enable Overwrite (OWREN=1) • Disable Write Left-Aligned (WLSIDE=0) • Select single shot (MODE=0) • Disable External Trigger (TRGEN=0) • Disable Injection Trigger (JTRGEN=0) • Enable BCTU triggering (BCTUEN=1) • Disable averaging (AVGEN=0) • Select conversion clock = 1/2 module clock (ADCCLKSEL=01)

Table continues on the next page...

Table 265. Example ADC configurations (continued)

Description	Settings
BCTU trigger mode	• Enable Overwrite (OWREN=1) • Disable Write Left-Aligned (WLSIDE=0) • Select single shot (MODE=0) • Disable External Trigger (TRGEN=0) • Disable Injection Trigger (JTRGEN=0) • Enable BCTU triggering (BCTUEN=1) • Enable all trigger sources (BCTU_MODE=1) • Disable averaging (AVGEN=0) • Select conversion clock = 1/2 module clock (ADCCLKSEL=01)

56.6.2 BCTU design considerations

For reasons discussed in ADC triggering, some applications require precise ADC triggering and BCTU provides that capability. The following topics discuss some design considerations when using BCTU.

56.6.2.1 BCTU or ADC data retrieval

Although the primary function of BCTU is to provide precisely timed triggers for ADC conversions, there is some overlap in BCTU and ADC function in that conversion data can be retrieved from either module. For applications in which frequent conversions are performed, such as real-time motor control, BCTU offers a distinct advantage over ADC, because BCTU trigger sources can be configured to route conversion results to a BCTU internal FIFO. Doing so prevents subsequent conversion data from overwriting previous data, making it available for either direct access or DMA transfer to system RAM.

56.6.2.2 Disabling BCTU triggers

There are fault scenarios in which an application must disable BCTU triggers as part of putting the system into a safe state. Examples include:

- Over-voltage condition
- Under-voltage condition
- Calibration parameter mismatch between the embedded motor control system and the motor

Application fault reaction design must include the disabling of BCTU triggers in scenarios in which either system conditions indicate an impending failure, or ADC conversion data is outside control boundaries.

56.6.3 eMIOS design considerations

In a BLDC motor control application, eMIOS plays a central role, providing the driving PWM signal as well as performing feedback signal measurements. The following topics discuss eMIOS considerations in the application design.

56.6.3.1 PWM considerations

In a BLDC motor control application, PWM/power signal requirements are affected by a number of parameters, including:

- Displacement Power Factor
- DF
- THD
- DVA

- CF
- FF

Ensure that the above parameters are included when choosing the timebase and prescaler combination for eMIOS channels producing PWM signals.

56.6.3.2 Input measurement

In a BLDC motor control application, eMIOS measures feedback signals for the motor and provides critical information to the control system. Selecting a timebase and prescaler for eMIOS channels performing input measurement tasks is usually a tradeoff between application requirements for precision and the computational cost in relation to the selected clock speed.

For example, some applications like electric steering demand an extremely precise control for position and torque. This is different from a compressor e-motor, where speed and torque do not demand rigid precision requirements.

Other considerations are related to control of:

- THD
- FF
- GM
- pf correction

The above considerations are significant from the perspective of the PWM reaction speed for the output signal for BLDC applications. These parameters must be considered in the design.

56.6.3.3 eMIOS channel modes useful in motor control

Table 266. eMIOS channel modes useful in motor control

Mode	Description	Use in motor control
MCB	Modulus Counter Buffered (Up/Down)	Time base • PWM frequency • Reload
IPWM	Input Pulse Width Measurement	Capture modes: • Speed measurement • Position measurement • Voltage (duty cycle)
IPM	Input Period Measurement	Speed measurement
PEC	Pulse Edge Counting	Position measurement
OPWMCB	Center aligned Output PWM Buffered with dead time	PWM duty cycle
OPWMB	Output Pulse Width Modulation Buffered	Trigger placement within the period
OPWMT	Output Pulse Width Modulation Trigger	Trigger placement within the period

56.6.3.3.1 MCB PWM and trigger generation

The following figure shows one option for PWM and trigger generation.

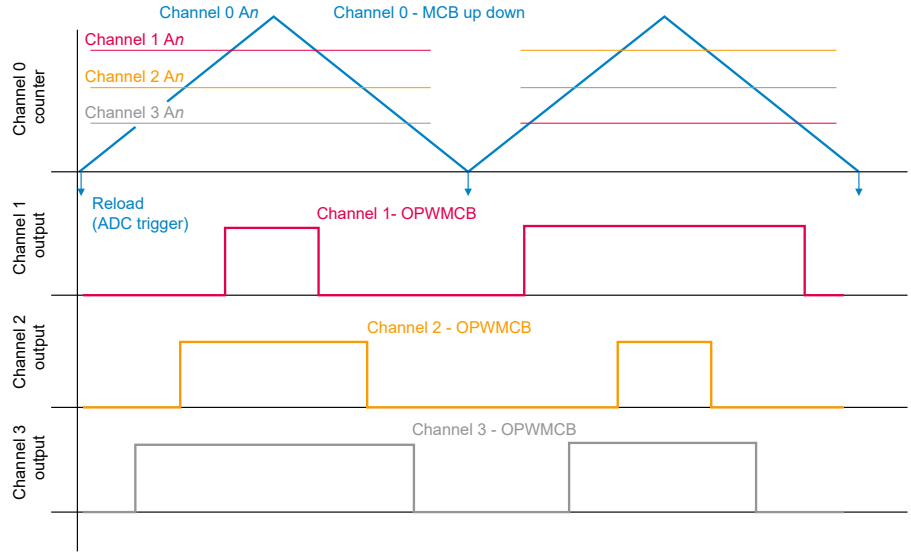


Figure 218. MCB PWM trigger generation

56.6.3.3.2 OPWMCB PWM and trigger generation

The following figure shows another option for PWM and trigger generation.

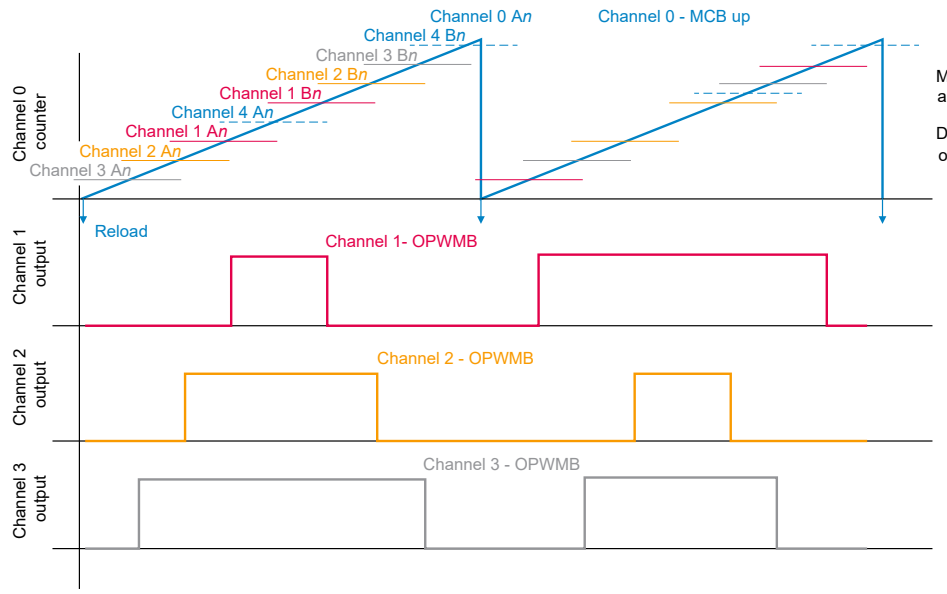


Figure 219. OPWMCB PWM trigger generation

56.7 Glossary

BLDC	Brushless DC. An electronically commutated DC motor.
CF	Crest Factor. A waveform parameter that is the ratio of peak values to its RMS value.
DF	Distortion Factor.

Displacement Power Factor	In power electronics, the phase relationship between voltage and current: $\cos(\theta_v - \theta_i)$, where θ_v is the phase of the voltage and θ_i is the phase of the current.
DVA	Distortion Volt-Amps.
FF	Form Factor.
FOC	Field-Oriented Control.
GM	Gain Margin.
pf	Power factor. In power electronics, it is the ratio of average power to RMS power.
PMSM	Permanent Magnet Synchronous Motor.
THD	Total Harmonic Distortion.
VFD	Variable-Frequency Drive.

Chapter 57
Analog-to-Digital Converter (ADC)

57.1 Chip-specific ADC information

57.1.1 ADC Configuration

This chip has up to 3 instances of ADC. ADC channels are divided into 3 groups - Precision, Standard and External (each with independent configuration settings and different accuracy/performance level). The following table shows the ADC configuration:

Table 267. ADC instances

Instance	MWCT2D17S	MWCT2D16S/MWCT2016S/MWCT2015S
ADC_0	Yes	Yes
ADC_1	Yes	Yes
ADC_2	Yes	No

Table 268. ADC configuration

Feature	ADC_0	ADC_1	ADC_2
No. of precision channels	8	8	8
No. of standard channels	16	16	16
No. of special internal channels	1[1]	0	0
No. of external channels	32	32	0
BCTU trigger support	Yes	Yes	Yes
DMA support	Yes	Yes	Yes
Hardware interleaving	Yes	Yes	Yes
No. of watchdogs	4	4	4
No. of Hardware trigger	3	3	3

1. CHAN_INT0 - channel 50(S)

Table 269. ADC maximum clock frequency

Feature	MWCT2D17S/ MWCT2D16S	MWCT2016S	MWCT2015S
ADC module clock frequency (f_mc)	160/120 MHz	120 MHz	120 MHz

NOTE

For register configurations need for different frequencies, see 'Functional description' section in this chapter.

ADC Channel mapping Table

Table 270. ADC channel mapping

Channel Number	ADC_0	ADC_1	ADC_2
0	ADC0_P0	ADC1_P0	ADC2_P0
1	ADC0_P1	ADC1_P1	ADC2_P1
2	ADC0_P2	ADC1_P2	ADC2_P2
3	ADC0_P3	ADC1_P3	ADC2_P3
4	ADC0_P4	ADC1_P4	ADC2_P4
5	ADC0_P5	ADC1_P5	ADC2_P5
6	ADC0_P6	ADC1_P6	ADC2_P6
7	ADC0_P7	ADC1_P7	ADC2_P7
8	—[1]	—	—
9	—	—	—
10	—	—	—
11	—	—	—
12	—	—	—
13	—	—	—
14	—	—	—
15	—	—	—
16	—	—	—
17	—	—	—
18	—	—	—
19	—	—	—
20	—	—	—
21	—	—	—
22	—	—	—
23	—	—	—
24	—	—	—
25	—	—	—
26	—	—	—
27	—	—	—
28	—	—	—
29	—	—	—
30	—	—	—
31	—	—	—

Table continues on the next page...

Table 270. ADC channel mapping (continued)

Channel Number	ADC_0	ADC_1	ADC_2
32	ADC0_S[8]	ADC1_S[8]	ADC2_S[8]
33	ADC0_S[9]	ADC1_S[9]	ADC2_S[9]
34	ADC0_S[10]	ADC1_S[10]	ADC2_S[10]
35	ADC0_S[11]	ADC1_S[11]	ADC2_S[11]
36	ADC0_S[12]	ADC1_S[12]	ADC2_S[12]
37	ADC0_S[13]	ADC1_S[13]	ADC2_S[13]
38	ADC0_S[14]	ADC1_S[14]	ADC2_S[14]
39	ADC0_S[15]	ADC1_S[15]	ADC2_S[15]
40	ADC0_S[16]	ADC1_S[16]	ADC2_S[16]
41	ADC0_S[17]	ADC1_S[17]	ADC2_S[17]
42	ADC0_S[18]	ADC1_S[18]	ADC2_S[18]
43	ADC0_S[19]	ADC1_S[19]	ADC2_S[19]
44	ADC0_S[20]	ADC1_S[20]	ADC2_S[20]
45	ADC0_S[21]	ADC1_S[21]	ADC2_S[21]
46	ADC0_S[22]	ADC1_S[22]	ADC2_S[22]
47	ADC0_S[23]	ADC1_S[23]	ADC2_S[23]
48	Bandgap	Bandgap	Bandgap
49	Tempsensor output	Tempsensor output	Tempsensor output
50	ANAMUX_OUT	—	—
51	—	—	—
52	—	—	—
53	—	—	—
54	VREFL	VREFL	VREFL
55	VREFH	VREFH	VREFH
56	—	—	—
57	—	—	—
58	—	—	—
59	—	—	—
60	—	—	—
61	—	—	—
62	—	—	—
63	—	—	—

Table continues on the next page...

Table 270. ADC channel mapping (continued)

Channel Number	ADC_0	ADC_1	ADC_2
64	ADC0_X[0] ADC0_MA[2:0]=3'h0	ADC1_X[0] ADC1_MA[2:0]=3'h0	—
65	ADC0_X[0] ADC0_MA[2:0]=3'h1	ADC1_X[0] ADC1_MA[2:0]=3'h1	—
66	ADC0_X[0] ADC0_MA[2:0]=3'h3	ADC1_X[0] ADC1_MA[2:0]=3'h3	—
67	ADC0_X[0] ADC0_MA[2:0]=3'h2	ADC1_X[0] ADC1_MA[2:0]=3'h2	—
68	ADC0_X[0] ADC0_MA[2:0]=3'h6	ADC1_X[0] ADC1_MA[2:0]=3'h6	—
69	ADC0_X[0] ADC0_MA[2:0]=3'h7	ADC1_X[0] ADC1_MA[2:0]=3'h7	—
70	ADC0_X[0] ADC0_MA[2:0]=3'h5	ADC1_X[0] ADC1_MA[2:0]=3'h5	—
71	ADC0_X[0] ADC0_MA[2:0]=3'h4	ADC1_X[0] ADC1_MA[2:0]=3'h4	—
72	ADC0_X[1] ADC0_MA[2:0]=3'h0	ADC1_X[1] ADC1_MA[2:0]=3'h0	—
73	ADC0_X[1] ADC0_MA[2:0]=3'h1	ADC1_X[1] ADC1_MA[2:0]=3'h1	—
74	ADC0_X[1] ADC0_MA[2:0]=3'h3	ADC1_X[1] ADC1_MA[2:0]=3'h3	—
75	ADC0_X[1] ADC0_MA[2:0]=3'h2	ADC1_X[1] ADC1_MA[2:0]=3'h2	—
76	ADC0_X[1] ADC0_MA[2:0]=3'h6	ADC1_X[1] ADC1_MA[2:0]=3'h6	—
77	ADC0_X[1] ADC0_MA[2:0]=3'h7	ADC1_X[1] ADC1_MA[2:0]=3'h7	—
78	ADC0_X[1]	ADC1_X[1]	—

Table continues on the next page...

Table 270. ADC channel mapping (continued)

Channel Number	ADC_0	ADC_1	ADC_2
	ADC0_MA[2:0]=3'h5		
79	ADC0_X[1] ADC0_MA[2:0]=3'h4	ADC1_X[1] ADC1_MA[2:0]=3'h4	—
80	ADC0_X[2] ADC0_MA[2:0]=3'h0	ADC1_X[2] ADC1_MA[2:0]=3'h0	—
81	ADC0_X[2] ADC0_MA[2:0]=3'h1	ADC1_X[2] ADC1_MA[2:0]=3'h1	—
82	ADC0_X[2] ADC0_MA[2:0]=3'h3	ADC1_X[2] ADC1_MA[2:0]=3'h3	—
83	ADC0_X[2] ADC0_MA[2:0]=3'h2	ADC1_X[2] ADC1_MA[2:0]=3'h2	—
84	ADC0_X[2] ADC0_MA[2:0]=3'h6	ADC1_X[2] ADC1_MA[2:0]=3'h6	—
85	ADC0_X[2] ADC0_MA[2:0]=3'h7	ADC1_X[2] ADC1_MA[2:0]=3'h7	—
86	ADC0_X[2] ADC0_MA[2:0]=3'h5	ADC1_X[2] ADC1_MA[2:0]=3'h5	—
87	ADC0_X[2] ADC0_MA[2:0]=3'h4	ADC1_X[2] ADC1_MA[2:0]=3'h4	—
88	ADC0_X[3] ADC0_MA[2:0]=3'h0	ADC1_X[3] ADC1_MA[2:0]=3'h0	—
89	ADC0_X[3] ADC0_MA[2:0]=3'h1	ADC1_X[3] ADC1_MA[2:0]=3'h1	—
90	ADC0_X[3] ADC0_MA[2:0]=3'h3	ADC1_X[3] ADC1_MA[2:0]=3'h3	—
91	ADC0_X[3] ADC0_MA[2:0]=3'h2	ADC1_X[3] ADC1_MA[2:0]=3'h2	—
92	ADC0_X[3] ADC0_MA[2:0]=3'h6	ADC1_X[3] ADC1_MA[2:0]=3'h6	—

Table continues on the next page...

Table 270. ADC channel mapping (continued)

Channel Number	ADC_0	ADC_1	ADC_2
93	ADC0_X[3] ADC0_MA[2:0]=3'h7	ADC1_X[3] ADC1_MA[2:0]=3'h7	—
94	ADC0_X[3] ADC0_MA[2:0]=3'h5	ADC1_X[3] ADC1_MA[2:0]=3'h5	—
95	ADC0_X[3] ADC0_MA[2:0]=3'h4	ADC1_X[3] ADC1_MA[2:0]=3'h4	—

1. Channels marked with “-“ means they are reserved and access to any reserved channel may result in inappropriate data.

NOTE

The decoding signals for the selection of external ADC channels (ADC_X) are gray-encoded.

NOTE

Value of ANAMUX_OUT is selected by DCMRWF1[SUPPLY_MON_SEL] field.

NOTE

The module clock mentioned in this chapter is CORE_CLK for this chip.

57.1.2 ANAMUX for internal supply monitoring

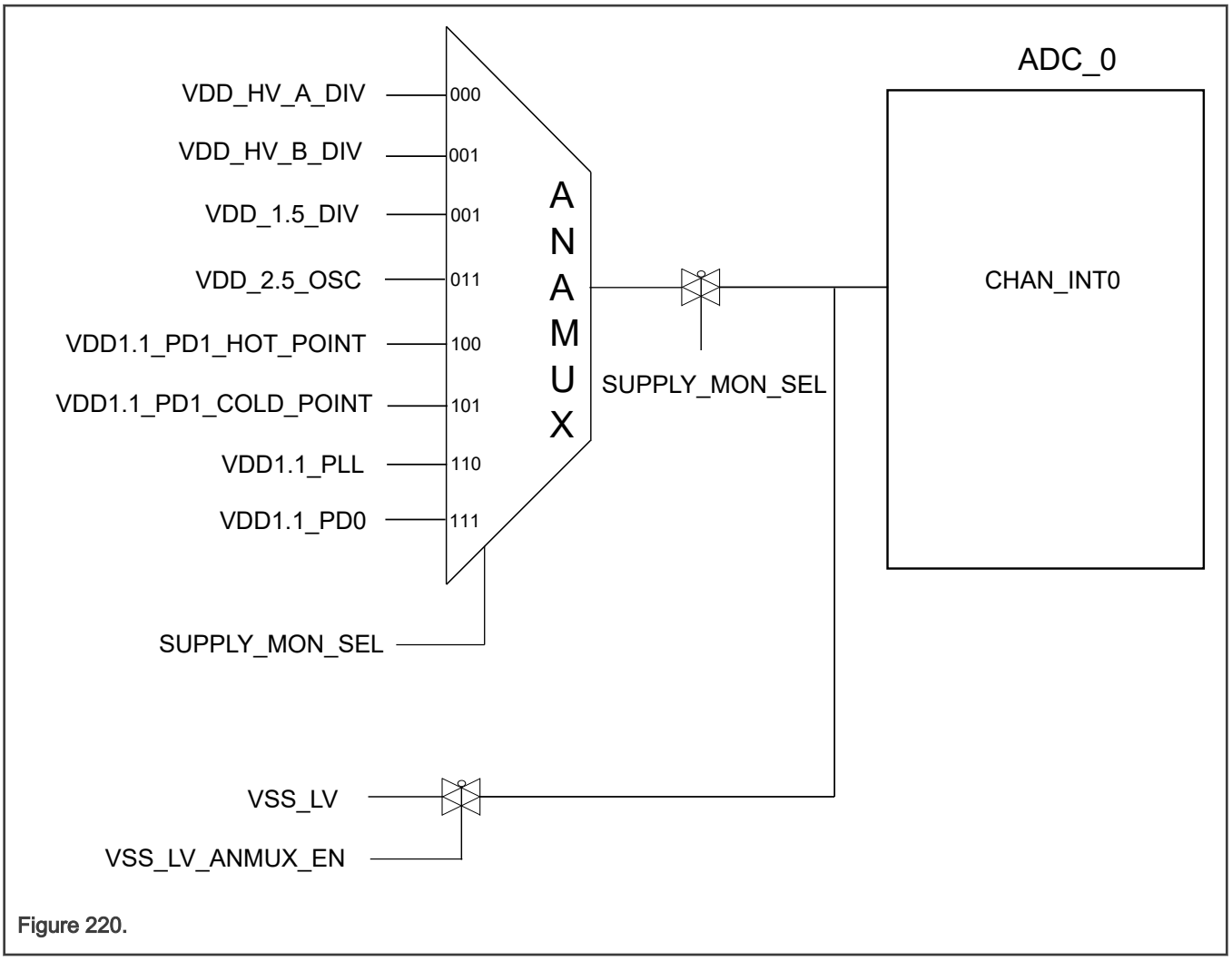


Figure 220.

NOTE

VDD_HV_B_DIV, VDD_1.5_DIV are not available in MWCT2016S and MWCT2015S.

57.1.3 BCTU Interface

All 3 instances of ADC are triggered from the BCTU. The BCTU provides channel conversion commands to the ADC (includes channel number information). In addition, the ADC provides the conversion result back to the BCTU.

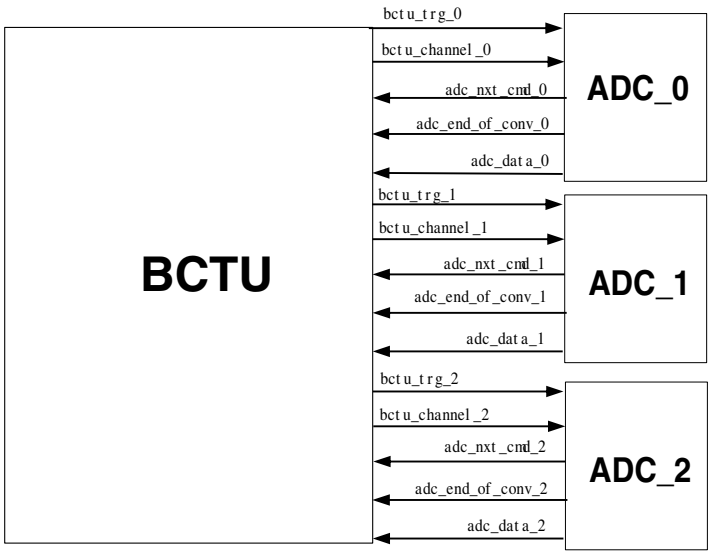


Figure 221. ADC Cross-triggering Unit

BCTU gives trigger pulse to all the 72 channels of ADC to initiate a conversion based on BCTU channel number.

BCTU-ADC result register

ADC result registers (ADCx_PCDRn[CDATA], ADCx_ICDRn[CDATA] and ADCx_ECDRn[CDATA]) store 15 bit conversion data. The BCTU ADC registers store 15-bit conversion data (BCTU_ADC0DR[ADC0_DATA], BCTU_ADC1DR[ADC1_DATA] and BCTU_ADC2DR[ADC2_DATA]).

57.1.4 DMA interface

Each ADC supports a single DMA request that can be requested after the conversion of every channel. The conversion result is stored into a register DMAR and is transferred via DMA or host access.

NOTE

The last ongoing conversion will be aborted as soon as ADC gets STOP request from the software and it will move to STOP mode.

57.1.5 Hardware triggers

In this chip, hardware trigger signals can be provided from TRGMUX outputs. This feature enables synchronous conversion of two independent ADC instances in parallel. If ADC is in Idle state (that is, no conversion phase ongoing and the MCR[PWDN] and MCR[ACKO] fields are 0) and the MCR[XSTRTEN] is set, an event on the external start signal causes ADC to start either normal or injected conversion operation. The MCR[NSTART]/MCR[JSTART] field is automatically set respectively. The normal conversion sync pulse is used with normal conversion trigger to synchronize the start timing of normal conversion between multiple ADC instances.

Table 271. Hardware triggers

TRGMUX output number	Hardware trigger functions
0	ADC_0 Normal Conversion
1	ADC_0 Injected Conversion
2	ADC_0 Normal Conversion Sync Pulse

Table continues on the next page...

Table 271. Hardware triggers (continued)

TRGMUX output number	Hardware trigger functions
4	ADC_1 Normal Conversion
5	ADC_1 Injected Conversion
6	ADC_1 Normal Conversion Sync Pulse
8	ADC_2 Normal Conversion
9	ADC_2 Injected Conversion
10	ADC_2 Normal Conversion Sync Pulse

57.1.6 ADC Self-Test

It is important to check at regular intervals if the ADC is operating correctly. For this purpose a self test feature is provided. When self-test is enabled, the ADC automatically checks its components and flags errors.

57.1.7 ADC mux-mode channels

There are some channels in ADC which are driven by multiple pads. For this, GPR bits from DCM are used to define which pad is connected to a particular ADC channel.

For example as below, where ADC0 S8 is driven by GPIO_PAD0 (default) and GPIO_PAD45 (mux_mode). ADC2 S8 is driven by GPIO_PAD133 (default) and GPIO_PAD45 (mux_mode). ADC1 S8 has a dedicated pad GPIO45, so its TGATE doesn't need GPR bit gating. Similar implementation is for ADC0_S9 and ADC2_S9. Following DCM GPR bits have been used:

Table 272. ADC mux_mode GPR

ADC channel	GPR_bit	Connected pad
ADC0 Standard channel 8 (ADC0 S8)	DCM.DCMRWF4[1]	0: GPIO PAD0 (default) 1: GPIO PAD45
ADC0 Standard channel 9 (ADC0 S9)	DCM.DCMRWF4[2]	0: GPIO PAD1 (default) 1: GPIO PAD46
ADC 1 standard channel 14 (ADC1 S14)	DCM.DCMRWF4[3]	0: GPIO PAD69 (default) 1: GPIO PAD32
ADC 1 standard channel 15 (ADC1 S15)	DCM.DCMRWF4[4]	0: GPIO PAD4 (default) 1: GPIO PAD33
ADC1 Standard channel 22 (ADC1 S22)	DCM.DCMRWF4[5]	0: GPIO PAD124 (default) 1: GPIO PAD145
ADC1 Standard channel 23 (ADC1 S23)	DCM.DCMRWF4[6]	0: GPIO PAD125 (default) 1: GPIO PAD146
ADC2 Standard channel 8 (ADC2 S8)	DCM.DCMRWF4[9]	0: GPIO PAD133 (default) 1: GPIO PAD45
ADC2 Standard channel 9 (ADC2 S9)	DCM.DCMRWF4[10]	0: GPIO PAD132 (default) 1: GPIO PAD46

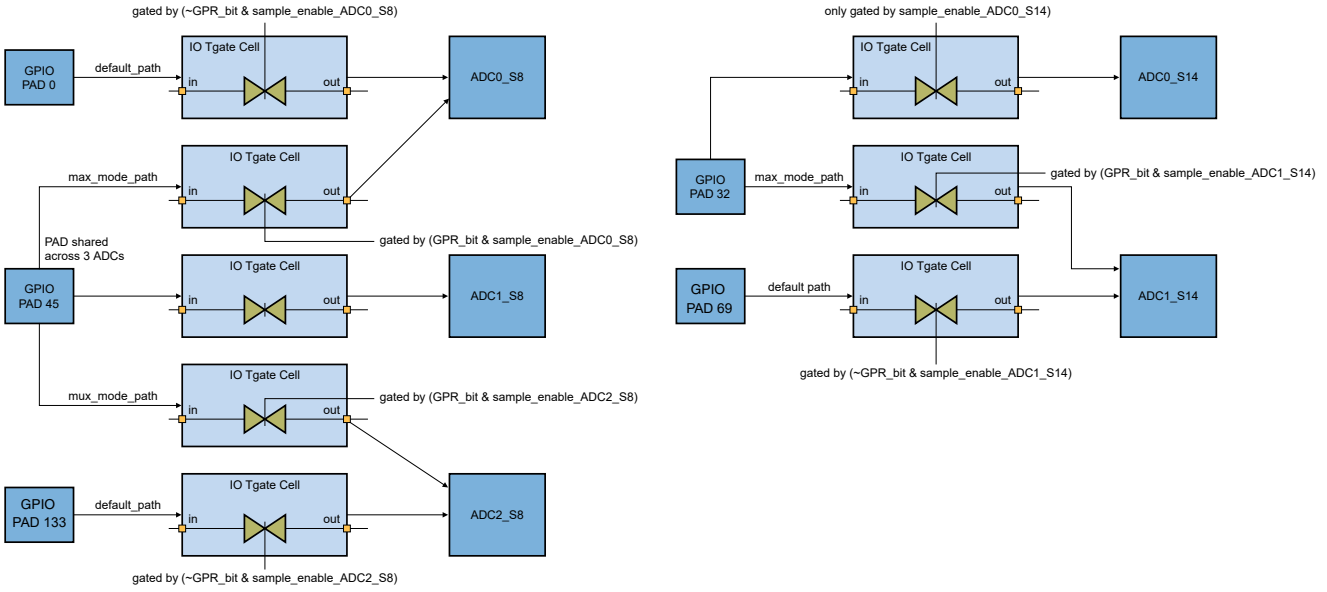


Figure 222. ADC channel interleaving

57.2 Introduction

ADC produces a digital value from an input analog voltage using the SAR algorithm. It features various methods to trigger a conversion and offers flexibility in the selection of input channels. ADC is useful for a wide range of applications.

A configurable self-test capability supports use cases with increased requirements for functional safety.

57.2.1 Features

- Selectable resolution (8-, 10-, 12-, 14-bit). Note that the conversion result is always 15 bits wide, even though the selected resolution is smaller (see CALBISTREG[RESN])
- Conversion data captured in a separate register for each input channel.
- Option to improve accuracy via averaging, which calculates conversion data by averaging the data of up to 32 conversions.
- Conversion triggers:
  - Normal conversion trigger converts a number of input channels, either once per trigger or continuously.
  - Injected conversion trigger interrupts an ongoing normal conversion and converts another set of input channels.
  - BCTU conversion trigger interrupts an ongoing conversion and converts an input channel, in which the input is selected and the conversion is started via the BCTU.
- An analog watchdog optionally monitors conversion data for each input channel and issues an interrupt if the converted data is below or above configurable limits.
- DMA functionality transfers conversion data to other modules.
- Programmable interrupts optionally issue an interrupt when conversion of one or of a set of input channels is finished.
- Self-test functions validate ADC structural integrity during functional operation and generate events with different severities on any finding.
- Conversion clock (AD_clk) control enables the use of ADC in systems with a higher clock frequency by using internal clock dividers.
- Auto turn-off of the conversion clock when ADC is idle.

57.2.2 Block diagram

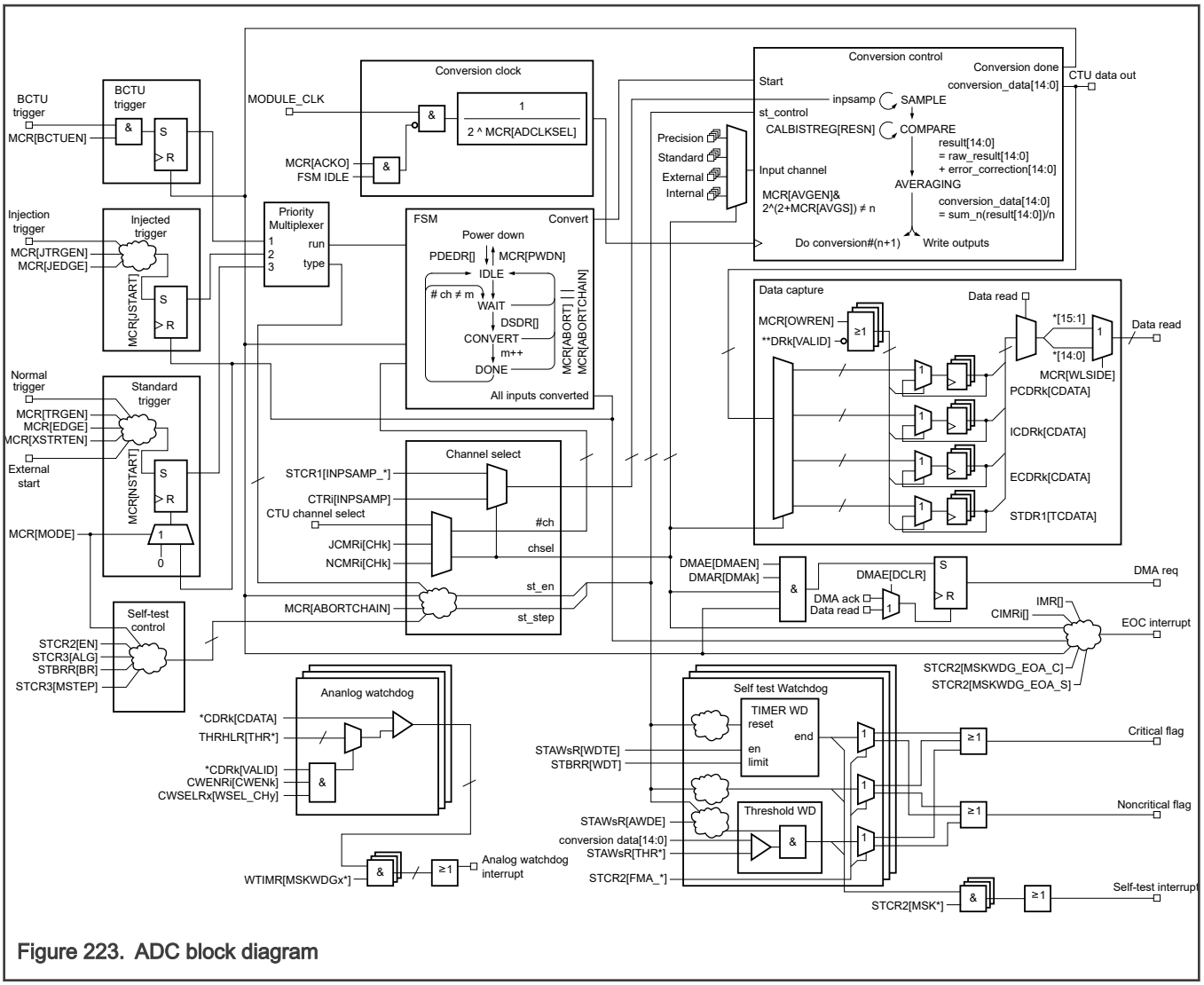


Figure 223. ADC block diagram

Guide to the diagram:

- The signal flow through the diagram is from left to right: inputs to blocks are on the left side, outputs are on the right.
- Block names have the first letters in uppercase and the other letters in lowercase—"Conversion Control," for example).
- Register names are in uppercase followed by a field name in brackets ("MCR[NSTART]," for example). Letters in lowercase are indexes. Asterisks are placeholders for letters (wildcards). Register names in the block diagram map to the names in ADC register descriptions.
- Signal names are in lowercase ("conversion done," for example).
- All digital signals shown are synchronous signals. Asynchronous inputs to the ADC are internally synchronized (synchronizers not shown).

The block diagram shows the signal flow through ADC from the left to the right and from the top to the bottom:

1. When ADC receives one of the three different types of triggers (BCTU, injected, or normal), the priority multiplexer initiates the conversion.
2. The input to be converted is selected in the Channel Select subblock, based on the type of trigger and on the self-test configuration.

3. The state machine (FSM) transitions from Idle state to Convert state. In case the channel to convert changed since the last conversion and this new channel is an external channel, the conversion starts after a delay configured by DSDR[DSD].
4. After a conversion begins, the Conversion Control block holds the internal capacitance network CDAC in the sample phase for a time defined by CTRi[INPSAMP].
5. During the compare phase, CALBISTREG[RESN] defines the number of steps used to execute the successive approximation algorithm. The error_correction values, determined during calibration, are added to the raw result. If averaging is enabled (MCR[AVGEN] = 1), up to 32 (specified via MCR[AVGS]) conversion results are averaged to obtain the result.
6. When the conversion completes and all averaging steps have completed:
    a. The conversion done signal indicates completion.
    b. Conversion_data[14:0] is written to the result registers in the Data Capture subblock.
    c. The FSM transitions into Done state.
7. The next input in the Channel Select block is selected and the next conversion is started by the transition of the FSM to Convert state. This continues until all selected inputs have been converted.
8. If an analog watchdog is enabled, the conversion result is compared to the configured threshold. If the limits are exceeded, an interrupt can be triggered.
9. Self-test can check the integrity of ADC, either interleaved with normal conversions or as a standalone check.
10. If an ongoing conversion is aborted (by writing 1 to MCR[ABORT]) or an ongoing conversion of a set of input channels is aborted (by writing 1 to MCR[ABORTCHAIN]), the FSM transitions into Idle state. The Conversion Control block stops the current conversion if MCR[ABORT] was written with 1, or finishes the current conversion if MCR[ABORTCHAIN] was written with 1. In both cases, the conversion done output signal of the Conversion Control block is set to 1.

Some details are not shown in the block diagram:

- Calibration must be performed to determine the error_correction values (see Calibration) before ADC can do meaningful conversions.
- An input can presample the internal low or high reference voltages to avoid a memory effect that may be present in the comparison due to a previous conversion (see Presampling).
- Monitor ADC status via status flags.
    - See Main Status (MSR) for the FSM state.
    - See Interrupt Status (ISR) to monitor interrupts.
    - See Channel End Of Conversion Flag For Precision Inputs (CEOCFR0) for conversion phase per input.
    - See the following registers to monitor analog watchdogs:
        - Analog Watchdog Out Of Range For Precision Inputs (AWORR0)
        - Analog Watchdog Out Of Range For Standard Inputs (AWORR1)
        - Analog Watchdog Out Of Range For External Inputs (AWORR2)

      See Analog Watchdog Threshold Interrupt Status (WTISR) to monitor analog watchdog interrupts.
    - See Self-Test Status 1 (STSR1) – Self-Test Status 4 (STSR4) to monitor self-test.

57.2.3 Modes of operation

ADC is always in Functional mode. No other mode selection exists.

- Put ADC into Power Down state by writing 1 to MCR[PWDN] to reduce power consumption.
- Gate the clock signal by writing 1 to MCR[ACKO] when ADC is in Idle state.

57.3 Functional description

57.3.1 Clock

ADC is controlled by one clock signal, the module clock. Internally, the conversion circuit is controlled by the conversion clock, which is derived from the module clock.

The frequency of the conversion clock has to be within the limits defined in the data sheet. If the module clock frequency is higher than the maximum frequency of the conversion clock allowed during the functional conversion or during the calibration (see the chip data sheet), then you must configure the ADC conversion clock divider (MCR[ADCLKSEL]) so that the frequency of the conversion clock is within allowed limits. For some conversion clock frequencies, you must write 3d to AMSIO[HSEN]. See Table 273.

Table 273. Clock configuration for highest conversion speeds

Module Clock Frequency f_mc[1]	Configuration during calibration	Configuration during functional conversion
40 MHz < f_mc ≤ 80 MHz	MCR[ADCLKSEL] = 1h AMSIO[HSEN] = 0h	MCR[ADCLKSEL] = 0h AMSIO[HSEN] = 0h
80 MHz < f_mc ≤ 120 MHz	MCR[ADCLKSEL] = 1h AMSIO[HSEN] = 3h	MCR[ADCLKSEL] = 0h AMSIO[HSEN] = 3h
120 MHz < f_mc ≤ 160 MHz	MCR[ADCLKSEL] = 2h AMSIO[HSEN] = 0h	MCR[ADCLKSEL] = 1h AMSIO[HSEN] = 0h
160 MHz < f_mc ≤ 240 MHz	MCR[ADCLKSEL] = 2h AMSIO[HSEN] = 3h	MCR[ADCLKSEL] = 1h AMSIO[HSEN] = 3h
240 MHz < f_mc ≤ 320 MHz	MCR[ADCLKSEL] = 3h AMSIO[HSEN] = 0h	MCR[ADCLKSEL] = 2h AMSIO[17] = 0h
320 MHz < f_mc ≤ 480 MHz	MCR[ADCLKSEL] = 3h AMSIO[HSEN] = 3h	MCR[ADCLKSEL] = 2h AMSIO[HSEN] = 3h

1. Lower resulting frequencies are allowed—see the chip data sheet.

When ADC is not converting (state machine is in Idle state), the conversion clock can be turned off (MCR[ACKO]).

57.3.2 Conversion

ADC measures the voltage of a signal—it converts an analog input value into a digital representation. When you read the conversion result, you must divide this conversion result by the number of codes (defined by selected conversion resolution in CALBISTREG[RESN]), and multiply it with the reference voltage value. This calculation provides the voltage value of the converted signal.

To convert an analog input voltage into its digital representation, ADC:

1. Receives a signal, referred to as a trigger, indicating it is to perform a conversion.
2. Captures the voltage of a signal via internal capacitances in a process referred to as sampling.
3. Compares the voltage to the reference voltage, using the SAR algorithm.

Before a conversion can be performed, you must select:

- The trigger source(s) to use (on the chip level)

- The input channel(s) to convert (NCMR0–2, JCMR0–2)

Additionally, you can tailor the conversion to meet your needs by changing the default configuration. For example, you can:

- Configure ADC to convert the selected input channel(s) only once or continuously in a loop (MCR[MODE]).
- Specify the duration that ADC samples the input signal (CTR0/1/2).
- Specify the resolution of the conversion result by selecting the number of bits (CALBISTREG[RESN]).
- Enable an analog watchdog to generate an interrupt when a conversion result falls outside a specified range (see Analog watchdog functions).
- Enable the DMA interface to send requests and receive acknowledgments (see DMA functionality).
- Check the operational integrity of ADC by running a self-test (see Self-test).

After power-up or a functional reset, ADC remains in Power Down state until enabled by writing 0 to MCR[PWDN]. After the system is powered on, you must run the calibration (see Calibration), before you can do a meaningful conversion with ADC.

Some configuration fields (listed below) are writeable only in Power Down state. If you intend to write to them, you must do so before exiting Power Down state.

- Select the conversion clock frequency to be within the allowed limits for a conversion.

57.3.3 Normal trigger

Select the input channels to convert by configuring the fields in NCMR0, NCMR1, and NCMR2.

These registers must be programmed before the start of conversion. You cannot reconfigure them until the conversion of all selected channels is complete. The sequence is always to convert the selected input channels in this order:

1. Start from the lowest input channel of the precision input channels.
2. Proceed through the standard input channels in ascending order.
3. End with the highest input channel of the external input channels.

57.3.3.1 Starting conversions in normal conversion mode

After programming the required fields of the following registers, you can start a conversion by writing 1 to MCR[NSTART]:

- Normal Conversion Enable For Precision Inputs (NCMR0)
- Normal Conversion Enable For Standard Inputs (NCMR1)
- Normal Conversion Enable For External Inputs (NCMR2)
- Main Configuration (MCR)

A hardware trigger can also start a normal conversion.

- If the external trigger is enabled (MCR[TRGEN] = 1), an external trigger enable is detected to start the conversion. The enable is checked only during start of conversion.
- Detection of an active edge defined by MCR[EDGE] (rising = 1, falling = 0) on an external trigger transitions MCR[NSTART] to 1 and starts the normal conversion.
- When MCR[TRGEN] = 1 (for example, when external trigger is enabled), conversion can be started by software.
- Any external trigger or software initiated conversion in normal conversion mode is ignored during any ongoing conversion chain.

NOTE

To have reliable operation there must be a gap of 3 clock cycles between configuration of NCMRx and the start of a conversion (by writing 1 to MCR[NSTART], or arrival of an external or internal hardware start trigger). ADC calculates the number of channels to be converted during this period.

57.3.3.2 Operation modes in normal conversions

Two operational modes are available during normal conversion:

- One-Shot mode
- Scan mode

For normal conversion, select the mode via MCR[MODE]. The first phase of the conversion process involves sampling the analog channel. The next phase is the conversion phase, curing which the sampled analog value is converted to digital as shown in the following figure.

Figure 224. Normal conversion flow

In One-Shot mode (MCR[MODE] = 0), a sequential conversion specified in the NCMR*n* mask registers is performed only once. At the end of each conversion, the digital result of the conversion is stored in the corresponding data register (CDR*n*).

For example: Channels A→B→C→D→E→F→G→H are present in a device where channels B→D→E are to be converted in One-Shot mode. Conversion begins with channel B, followed by conversion of channels D and E. At the end of conversion of channel E, the scanning of channels stops.

MSR[NSTART] automatically transitions to 1 when a normal conversion starts. At the same time, MCR[NSTART] is reset to zero by hardware, enabling software to program a new start of conversion in advance. In this case, the new requested conversion starts after completion of the running/current conversion. However, for correct functioning of the device, you should program MCR[NSTART] for the new conversion only after completion of the current conversion. The application can wait for an ECH interrupt prior to writing 1 to MCR[NSTART] again.

If an external trigger starts the conversion chain, ADC does not observe the trigger input until the conversion chain is finished. After the chain is finished, the next chain can be triggered by another hardware trigger edge.

In Scan mode operation (MCR[MODE]=1), a sequential conversion of N channels specified in the NCMR*n* registers is continuously performed. At the end of each conversion, the result is stored in the corresponding data register, as in One-Shot mode.

MSR[NSTART] automatically transitions to 1 when a normal conversion starts. Unlike One-Shot mode, MCR[NSTART] is not reset to 0 in Scan mode. It can be reset by software when you must exit Scan mode. In that case, ADC completes the current chain and after the last conversion, it resets MCR[NSTART] to 0.

For example: Channels A→B→C→D→E→F→G→H are present in the device where channels B→D→E are to be converted in Scan mode. MCR[MODE] = 1 selects Scan mode operation. Conversion starts from channel B followed by conversion of channels D→E. After conversion of channel E, the scanning of channel B starts followed by conversion of the channels D→E. This sequence repeats itself until MCR[NSTART] is reset to 0 by software.

If an external trigger starts a conversion, then MCR[NSTART] does not transition to 1. This scan chain can be stopped by writing 0 to MCR[NSTART]. The conversion stops when the ongoing chain is finished. Consequently, after it starts, the only way to stop Scan mode conversion is to write 0 to MCR[MODE].

End of conversion

In both modes, at the end of each conversion, an End of Conversion (EOC) interrupt is issued if enabled by the corresponding mask fields in CIMR*n* and IMR).

After conversion of all selected channels in NCMR*n* is complete, the conversion operation is considered finished. Then ISR[ECH] transitions to 1 and the End of Chain (ECH) interrupt is issued (if enabled in IMR[MSKECH]).

The corresponding channel field in the CEOCFR*n* register(s) is updated to indicate that data is available on data register CDR*n* of the respective channel.

If there is no channel selected in NCMR*n* and there is a start-of-conversion trigger, then ISR[ECH] is set to 1 and the End of Chain (ECH) interrupt is immediately issued (if enabled).

57.3.4 Injected trigger

An injected trigger enables you to convert a set of input channels although the standard trigger has already started conversion of another set of input channels. Each channel can be individually enabled by writing 1 to the corresponding field in JCMR0, JCMR1, or JCMR2.

The set of input channels of the injected trigger is converted only once. A continuous conversion of a set of input channels in a loop is only possible when started by a standard trigger. When an injected trigger is received, an ongoing conversion started by a standard trigger is interrupted. After the set of input channels of the injected trigger is converted once, the interrupted conversion of the set of input channels of the standard trigger resumes from the input channel that was interrupted, as shown in the following figure.

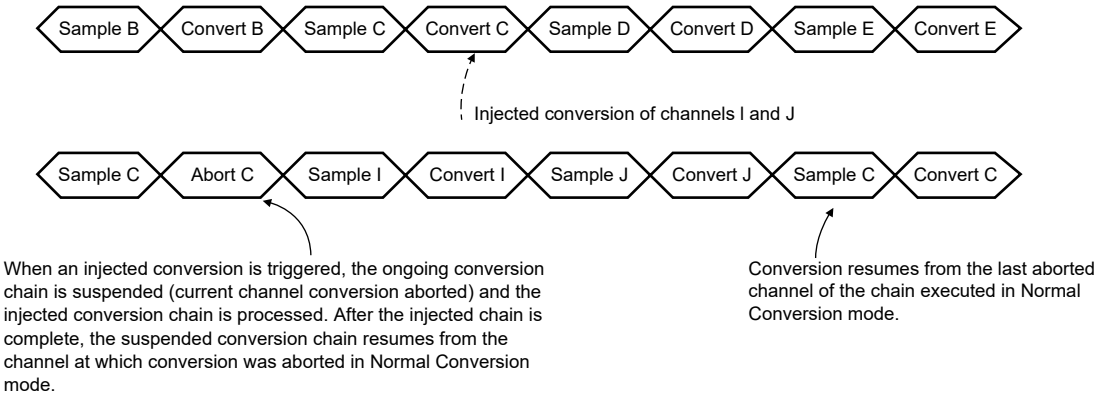


Figure 225. Injected trigger conversion sequence

57.3.4.1 Starting conversions in injected conversion mode

An injected conversion chain can be started:

- By software: When external triggering is disabled (MCR[JTRGEN] = 0), you can write 1 to MCR[JSTART], which causes the current conversion chain processed in Normal Conversion mode to be suspended and the injected chain to be processed.
- By external trigger: When external triggering is enabled (MCR[JTRGEN] = 1), a programmed event (rising/falling edge depending on the value of MCR[JEDGE]) on the injection external input starts the injected conversion.

MCR[JSTART] automatically transitions to 1 when the injected conversion chain starts. At the same time MCR[JSTART] is reset to 0, enabling software to program a new start of conversion in advance. In that case the new requested conversion starts after the running conversion completes.

At the end of each conversion, an End Of Injected Conversion (JEOC) interrupt is issued, if enabled by the corresponding mask field in Interrupt Mask (IMR). At the end of a chain, an End Of Injected Chain (JECH) interrupt is issued, if enabled by the corresponding mask field. Additionally, ADC writes 1 to ISR[JECH].

NOTE

If the content of all the Injected Conversion mask registers (JCMR*n*) is zero, that is, no channel is selected, the JECH interrupt is immediately issued after the start of conversion.

To have reliable ADC operation, there must be a gap of 3 cycles between configuration of JCMRx and starting of conversion (writing 1 to MCR[JSTART] or arrival of either an external or internal hardware start trigger). ADC calculates the total channel numbers to be converted during this period.

The corresponding channel bit in the CEOCFR*n* register is updated to indicate data is available in the channel's conversion data register, CDR*n*.

After starting, an injected chain conversion cannot be interrupted.

57.3.5 BCTU interface

The BCTU interface enhances ADC's injected conversion capability. It contains control inputs to select the channels to be converted from the appropriate event configuration register. Figure 226 shows the interface.

The BCTU generates a trigger (bctu_trigger) and a channel number (bctu_numchannel) to be converted. A single channel is converted for each request. After performing the conversion, ADC returns the result on the bctu_dataout bus together with two output signals named bctu_nextcmd and bctu_push. The assertion of signal bctu_nextcmd means ADC is ready to accept the next trigger from BCTU. The bctu_push signal is asserted at the end of conversion, meaning that conversion is finished and the conversion result available at output bctu_dataout is valid.

The conversion result is also saved in the corresponding channel's data register and is compared with analog watchdog thresholds if requested.

The signals bctu_trigger, bctu_nextcmd and bctu_push are all of type single-cycle active-high-pulse in the ADC clock domain. The channel number provided from BCTU must be valid when bctu trigger is active-high. The result data from ADC is valid with bctu_push high.

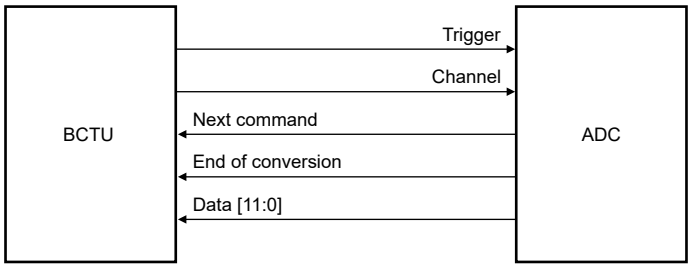


Figure 226. BCTU interface

The BCTU interface has two modes of operation:

- Trigger
- Control

To enable the BCTU interface, write 1 to MCR[BCTUEN]. The operating mode (Trigger or Control) can be fixed or programmable.

57.3.5.1 BCTU Trigger mode

In Trigger mode, normal and injected conversions can also be performed. All types of conversions can be initiated in this mode. The priorities among the three types of conversions are discussed below.

When a trigger is received, the channel number is taken as an injected channel value and the triggered injected conversion starts. MSR[BCTUSTART] transitions to 1 automatically at this point and it is also automatically reset to 0 when the triggered injected conversion is completed.

BCTU conversions must be requested only after successful ADC calibration. The application must prevent any BCTU trigger during calibration.

If a BCTU trigger is received during an ongoing injected conversion, the injected chain is aborted immediately and only the BCTU-triggered injected conversion proceeds. Additionally, MSR[JSTART] is reset to zero. The abort of an injected conversion is indicated by MSR[JABORT].

If a BCTU trigger is received during an ongoing normal conversion, the ongoing normal channel conversion is suspended and the BCTU-triggered injected conversion is processed. Normal conversion resumes from the suspended channel after completion of the BCTU-triggered conversion.

If a normal conversion is requested during BCTU conversion (MSR[BCTUSTART] = 1), the normal conversion starts after the BCTU conversion completes (MSR[BCTUSTART] is reset to zero).

Any injected conversion is discarded if requested during BCTU conversion and MCR[JSTART] is immediately reset to zero.

57.3.5.2 BCTU Control mode

In BCTU control mode (MCR[BCTU_MODE] = 0), only the BCTU can start a conversion. All other trigger sources are ignored.

Along with BCTU trigger, the information provided with signal bctu_numchannel is taken as the channel number for injected channel and BCTU triggered conversion starts. MSR[BCTUSTART] transitions to 1 automatically at the start of the conversion and it remains 1 unless BCTU is disabled by writing 0 to MCR[BCTUEN].

The conversion must be requested (generating bctu_trigger) when calibration has finished successfully. If a BCTU trigger is received during calibration execution, calibration is stopped immediately in order to satisfy the BCTU request. Calibration fails in this case.

57.3.6 Aborting a conversion

Two abort functions are provided:

- Abort of a single channel
- Abort of a chain

To abort an ongoing conversion, write 1 to MCR[ABORT]. The current conversion aborts and conversion of the next channel of the chain begins immediately.

Depending on the current state of a conversion, an Abort action may take 1 to 4 cycles of the bus clock. At the start of any conversion, and at the end of a particular conversion when internal state counters are changing, an abort action is delayed by a maximum of 3 cycles to put all states in a stable condition.

To ensure that the abort action completes, do not change MCR[ABORT] for the next 3 cycles.

Avoid writing 1 to MCR[ABORT] along with, and within 3 cycles of, writing 1 to MCR[NSTART] or MCR[JSTART].

During an abort:

- MCR[NSTART] and MCR[JSTART] remain 1, if not in the last channel of the chain.
- MCR[ABORT] is reset to 0 when the channel is aborted.
- The EOC interrupt corresponding to the aborted channel is not generated. This behavior applies to normal and injected conversions. If the last channel of a chain is aborted, the end of chain is reported by generating an ECH interrupt.

It is possible to abort the current chain of conversions by writing 1 to MCR[ABORTCHAIN]. In this case, the behavior of ADC depends on the value of MCR[MODE] (One-Shot/Scan operation modes). If Scan operation mode is disabled, MCR[NSTART] is automatically reset to 0 along with MCR[ABORTCHAIN]. Otherwise, in Scan Operation mode, a new chain is started. The EOC interrupt of the current aborted conversion is not generated but an ECH interrupt is generated to signal the end of the chain.

NOTE

For a single channel chain, an ECH interrupt is not generated for an abort or abort chain.

When an abort chain is requested while an injected chain is running over a suspended normal chain, both the injected and normal chains are aborted, and both MCR[NSTART] and MCR[JSTART] are reset to 0.

57.3.7 Configuring ADC clock divider and sample time settings

You can scale the AD_clk frequency via MCR[ADCLKSEL]. ADCLKSEL can only be written in Power Down state (MCR[PWDN] = 1). Depending on the frequency of the module clock, there might be different settings necessary for functional conversion and for calibration. See the chip data sheet for ADC electrical characteristics.

Three conversion timing registers are used (CTR*n*) to support different sampling times for the different types of channels. An exception exists for the temperature sensor channel, which always uses the Conversion Timing For Standard Inputs (CTR1) value. See the register description for details.

57.3.8 Presampling

The presampling feature enables precharge or discharge of the ADC internal sample capacitor node to a defined level before sampling of the selected analog input channel starts. This is useful for resetting information (history effect/offset) from the most recent conversion. During presampling, ADC samples the internally generated voltage. During the sampling phase, ADC samples analog input coming from pads.

Enable presampling on a per-channel basis by setting the corresponding fields in the PSR*n* registers to one.

After enabling presampling for a channel, the normal sequence of operation for the channel is:

1. Presampling
2. Sampling
3. Evaluation

You can bypass sampling of all channels by writing 1 to PSCR[PRECONV]. When sampling of a channel is bypassed, the sampled and stored internal voltage applied during the presampling phase is converted (Figure 227). See Conversion time for the timing equation for conversion.

Presampling does not apply for self-test channels.

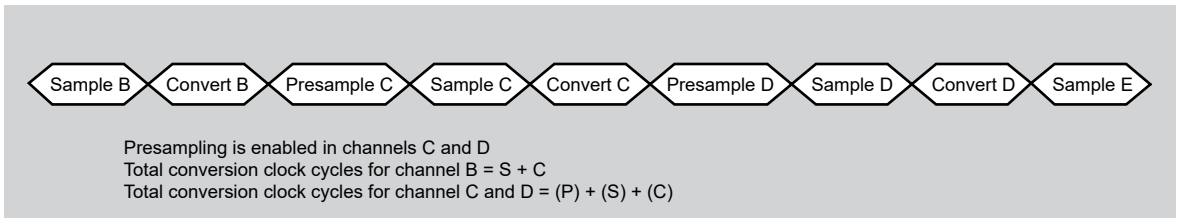


Figure 227. Presampling sequence

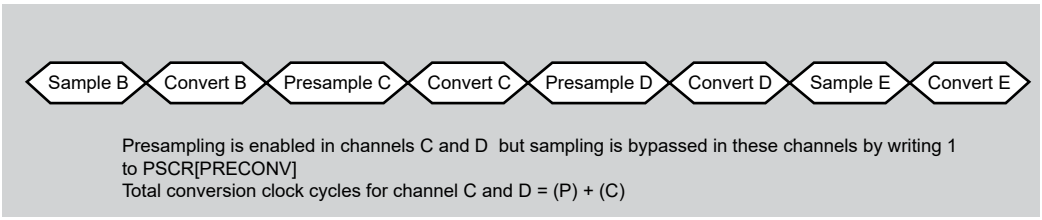


Figure 228. Presampling sequence with PRECONV=1

57.3.8.1 Enabling presampling per channel

Enable presampling for specific channels by programming Presampling Control (PSCR). You can select between two internally generated voltages by configuring the PREVAL*n* fields in Presampling Control (PSCR) to the appropriate values, as given in the following table.

Table 274. Presampling voltage selection via PREVAL*n*

PREVAL*n*	Presampling voltage
0	Presample voltage – 0: VREFL
1	Presample voltage – 1: VREFH

The presampling configuration field(s) (PSCR[PREVAL0], PSCR[PREVAL1], PSCR[PREVAL2]) are located in Presampling Control (PSCR). These fields enable selection of the presampling channel.

The temperature sensor channel has an exception. This channel is mapped with a fixed channel depending on device configuration, but it uses the presampling configuration defined for channel group 2. So, when the temperature sensor channel is selected, the presampling voltage defined by PSCR[PREVAL1] is selected. This voltage might be different from the one selected for channel group 1 via PSCR[PREVAL0].

57.3.9 Analog watchdog functions

When enabled, an analog watchdog monitors conversion results for an associated channel and reports an issue if it determines a result is outside customer-defined limits (as shown in Analog watchdog configuration (up to 16 watchdogs)). These limits are specified by an upper and a lower threshold value named THRH and THRL respectively.

After the conversion of the selected channel a comparison is performed between the converted value and the threshold values. If the converted value is outside that threshold value, then ADC generates a corresponding threshold violation interrupt.

The comparison result is stored as HAWIFx and LAWIFx fields in Analog Watchdog Threshold Interrupt Status (WTISR), as explained in the following table. Depending on the values of WTIMR[LAWIFENx] and WTIMR[HDWIFENx] mask fields, an interrupt is generated on a threshold violation.

Table 275. Values of HAWIFx and LAWIFx

HAWIFx	LAWIFx	Conversion result
1	0	Conversion result > THRH
0	1	Conversion result < THRL
0	0	THRH ≥ conversion result ≥ THRL

Depending on the chip's factory settings, up to 16 analog watchdogs are available. See the chip-specific configuration information for availability.

There are two types of mutually exclusive settings for analog watchdogs. Both types are described in the following sections. The availability depends on device configuration. See the chip-specific configuration information for availability.

Different capabilities are available with the two settings, as described below.

57.3.9.1 Channel analog watchdog select registers

ADC configuration includes selecting the analog watchdog threshold register (THRHLR*n*) that provides limits to monitor inputs of a given type. You select the THRHLR*n* using CWSELR*a*l*n* registers.

Table 276. CWSELR*a*l*n* register naming

Index	Description
a	ADC input type associated with a CWSELR*a*l*n* register. • P = Precision inputs • S = Standard inputs • E = External inputs
n	0-indexed register number.

57.3.9.2 Analog watchdog configuration (up to 16 watchdogs)

ADC can have up to 16 watchdogs in the factory configuration. (See the chip-specific configuration information for availability.) For each watchdog there is one threshold value register, THRHLR*n*, that contains the high and low thresholds.

You can independently enable the analog watchdog for each channel by programming the appropriate CWENR*n* register.

The threshold values for each channel can be selected independently from a maximum of 16 threshold registers (THRHLR*n*) using the corresponding WSEL_CH*n* field in the appropriate CWSELR*a*l*n*, where *a* indicates the input type:

- P = precision
- S = standard
- E = external

Each CWSELR*a*l*n* register is associated with 8 consecutive channels. CWSELR*a*l0 holds 8 WSEL_CH fields for channels 0 to 7. CWSELR*a*l1 is for channels 8 to 15 and so on. The availability of fields and registers depends on device configuration. See the chip-specific configuration information for availability.

If the conversion result of a selected channel is outside the range specified by threshold values, then the corresponding field in the Analog Watchdog Out of Range register (AWORR*n*) transitions to 1. For example, if channel 7 is to be monitored with the threshold values in THRHLR3, then CWSELR*a*l0[WSEL_CH7] must be programmed with 3. Enabling the watchdog is done by writing 1 to the corresponding field for channel 7 in CWENR0.

In this configuration, a set of threshold values (THRHLR*n*) can be linked to several ADC channels. The threshold values to be selected for a channel must be programmed only once in CWSELR*a*l*n*,.

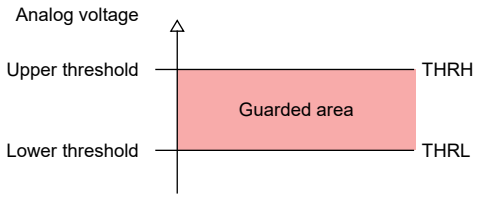


Figure 229. Guarded area

NOTE

If the higher threshold of the analog watchdog is programmed below the lower threshold and the conversion result is below the lower threshold, then the WDG*n*L interrupt for the low threshold violation transitions to 1. If the conversion result is greater than the lower threshold (and therefore also greater than the higher threshold), then the interrupt WDG*n*H for high threshold violation is generated. Thus you should take care to avoid that situation, as it could lead to misinterpretation of the watchdog interrupts.

57.3.10 DMA functionality

Conversion result data from any channel can be transferred from a register to system memory via Direct Memory Access (DMA). DMA transfers are enabled by writing 1 to DMAE[DMAEN]. After being enabled, the on-chip DMA controller can receive a DMA request after the conversion of each channel by writing 1 to the respective masking field in DMA Request Enable For Precision Inputs (DMAR0), DMA Request Enable For Standard Inputs (DMAR1), or DMA Request Enable For External Inputs (DMAR2). DMA masking registers must be programmed before starting any conversion.

A DMA request to the DMA controller can be cleared at different times in two modes:

- Mode-1: Clearing of DMA request on acknowledgment from DMA controller (DMAE[DCLR] = 0)
- Mode-2: Clearing of DMA request on read to data registers (DMAE[DCLR] = 1)

The figures below show the operation of DMA in two modes (cycle counts are typical values).

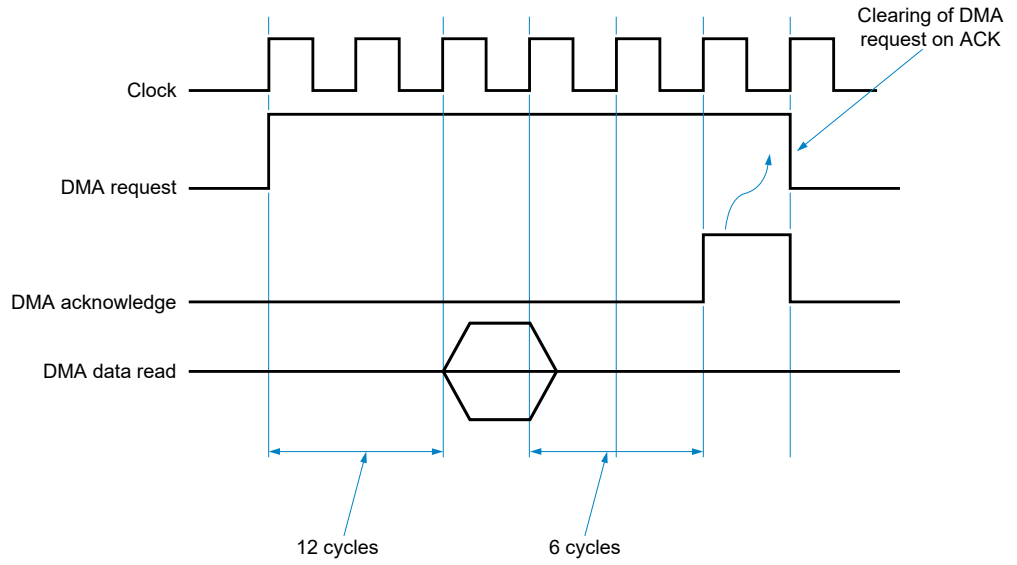


Figure 230. DMA operation Mode-1 (DMAE[DCLR] = 0)

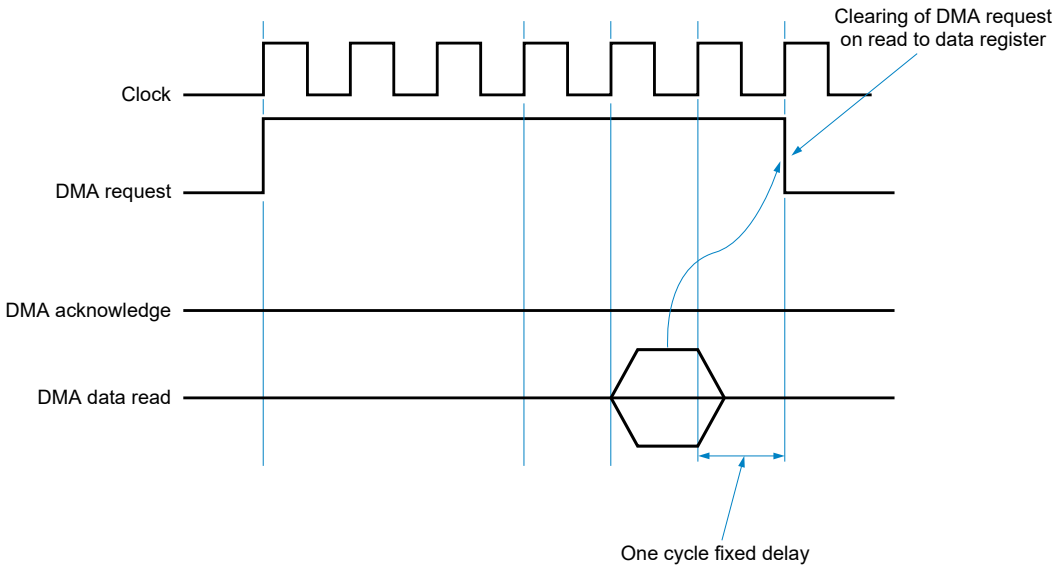


Figure 231. DMA operation Mode-2 (DMAE[DCLR] = 1)

57.3.11 Interrupts

ADC generates the following maskable interrupts:

- EOC (End of Conversion)
- ECH (End of Chain)
- JEOC (End of Injected Conversion)
- JECH (End of Injected Chain)
- EOBCTU (End of BCTU-triggered Conversion)
- LAWIFx and HDWIFx (Watchdog threshold)

- Self-testing interrupts

Interrupts are generated during the conversion process to signal events such as End Of Conversion (EOC), and so on, as described in Interrupt Status (ISR). ISR and another register, Interrupt Mask (IMR), are provided to check and enable the interrupt request to an external interrupt controller.

Interrupts can be individually enabled on a per-channel basis by programming EOC Interrupt Enable For Precision Inputs (CIMR0).

CEOCFR*n* are used for pending End of Conversion interrupts, with one field per channel to indicate completed conversions on a per-channel basis.

Interrupts generated due to the analog watchdogs are managed by two 32-bit registers, Analog Watchdog Threshold Interrupt Status (WTISR) and Analog Watchdog Threshold Interrupt Enable (WTIMR), to check and enable the interrupt request. A watchdog interrupt causes two corresponding fields to transition to 1 for each channel monitored. One field is in WTISR[WDG*n*H] and another in WTISR[WDG*n*L].

Generated events are logged into status registers and used to generate interrupts if enabled in corresponding Mask registers.

There are three different interrupt outputs. The interrupt structure is shown in Figure 232.

Each interrupt is generated by combining a group of events.

The following interrupts are combined (logic OR) into one interrupt output.

- EOC
- ECH
- EOBCTU
- JEOC
- JECH
- End of self-test algorithm (WDG_EOA_S and WDG_EOA_C)

Interrupts of all HDWIFx and LAWIFx are combined on the second interrupt output.

Interrupts of all self-test errors (watchdog threshold, sequence, and timer violations) in Self-Test Status 1 (STSR1) are combined on the third interrupt output. The errors are:

- WDSERR
- WDTERR
- ERR_S0
- ERR_S1
- ERR_S2
- ERR_C

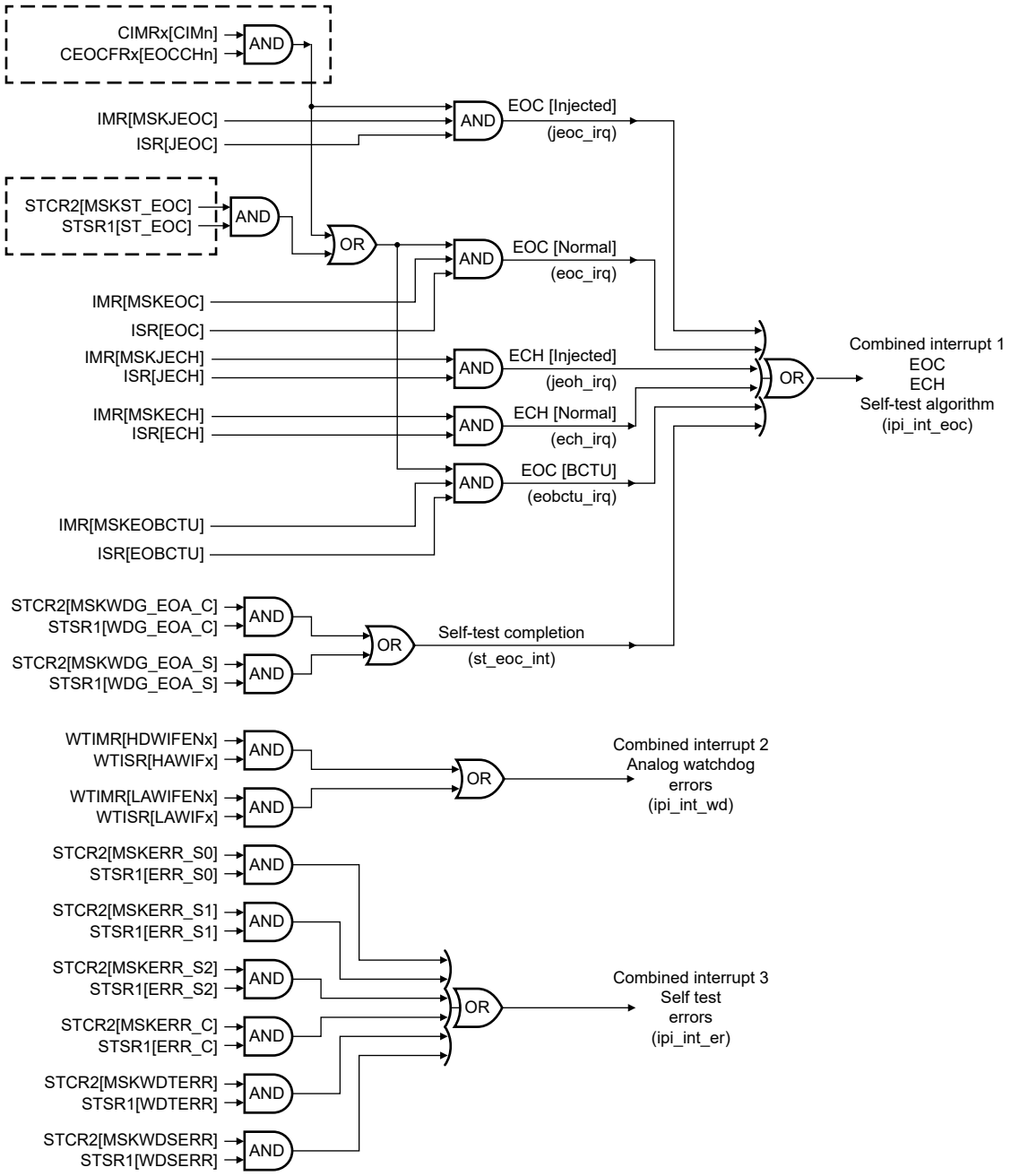


Figure 232. Interrupt structure of ADC

Interrupt Status (ISR) Indicates the interrupt pending request status. Write 1 to the corresponding status field to clear the pending interrupt flag (at this write operation all the other fields in Interrupt Status (ISR) must be maintained at 0).

ISR checks for three types of interrupts.

The first type, Interrupt 1, relates to the following:

- End of normal or injected conversion (EOC or JEOC)
- End of normal or injected conversion chain (ECH or JECH)
- End of algorithm C or algorithm S self-test (WD_EOA_C or WD_EOA_S)

Perform the following actions for this type of interrupt:

1. Read Interrupt Status (ISR) to determine whether the event is an EOC, JEOC, ECH, JECH, or EOBCTU event.
2. Read Self-Test Status 1 (STSR1) to identify the end of self-test algorithm events (WDG_EOA_S or WDG_EOA_C), if present.
3. If the interrupt is an EOC or JEOC event:
   - Clear the respective channel pending field in the appropriate CEOCFR*n* register for an EOC by writing a 1 to it.
   - Clear ISR[EOC] or ISRJEOC] by writing a 1 to it.
4. If the interrupt is an ECH or JECH, clear ISR[ECH] or ISR[JECH] by writing 1 to it.
5. If the interrupt is EOBCTU, clear ISR[EOBCTU] by writing 1 to it.
6. If the interrupt is for self-test end of algorithm, clear STSR1[WDG_EOA_S] or STSR1[WDG_EOA_C] by writing 1 to it.

The second type, Interrupt 2, relates to analog watchdog errors. Perform the following actions for this type of interrupt:

1. Read Analog Watchdog Threshold Interrupt Status (WTISR) to determine which watchdog triggered the interrupt.
2. Read the AWORRx register(s) to determine which channel's data caused the interrupt (if present).
3. Clear the respective field of the WTISR[WDG*n*H/WDG*n*L] register, where *n* = watchdog number, by writing 1 to it.
4. Clear the respective field of the AWORR*n*[AWOR_CH*n*] register, where x = register, n = channel, by writing 1 to it.

The third type, Interrupt 3, relates to self-test errors. Perform the following actions for this type of interrupt:

1. Read Self-Test Status 1 (STSR1) to determine the type of error encountered in self-test (WDSERR, WDTERR, ERR_S0, ERR_S1, ERR_S2, ERR_C, and so on) if present.
2. Clear the respective field in Self-Test Status 1 (STSR1) by writing 1 to it.

The connectivity of interrupt lines at the chip level (external to ADC) determines whether software must read all the registers for all three interrupts or any one or combination of any two.

57.3.12 External decode signals delay

The ADC provides three external decode signals which it uses to select one channel out of eight in the external analog multiplexers. There can be a maximum of four such multiplexers used to connect the 32 external channels. The ADC automatically sets the decode signals to control these external analog multiplexers, based on the current channel selected for conversion. Delay Start Of Data Conversion (DSDR) is provided to take the switching time of the external analog multiplexer into account. It enables you to program the delay between the decoding signal selection and the actual start of conversion.

If presampling is enabled, the DSD delay starts after the presample phase (presampling time is the same duration as sample time).

57.3.13 Power Down state

The analog part, along with the ADC controller, can be put in Power Down state by writing 1 to MCR[PWDN]. This shuts down ADC and stops the clock to the ADC controller. After release of reset, MCR[PWDN] remains 1, so the ADC analog module is kept in Power Down state by default. You must exit this state before starting any operation, by writing 0 to MCR[PWDN]. In Power Down state, no conversion can be started. If a BCTU trigger pulse is received during this state, it is discarded. You must ensure that a BCTU trigger is not initiated to ADC before writing 0 to MCR[PWDN], as it may cause the BCTU module to be stuck in a wait state.

You can set MCR[PWDN] by writing 1 to it at any time. If a conversion is ongoing, ADC does not move into Power Down state immediately. ADC enters Power Down state only after completion of the ongoing conversion. During scan operation, the ongoing operation should be aborted manually by writing 0 to MCR[NSTART] before or after writing 1 to MCR[PWDN].

ADC Power Down state status is indicated by MSR[ADCSTATUS] when ADC enters into Power Down state.

If the BCTU is enabled and MSR[BCTUSTART] = 1, then you cannot write 1 to PWDN. When BCTU Trigger mode is enabled, the application must wait for the end of conversion (MSR[BCTUSTART] is automatically reset to 0). When BCTU Control mode is enabled, before entering Power Down state the application must write 0 to BCTUEN also.

If Power Down state is entered by writing 1 to MCR[PWDN], the process running before entry into Power Down state must be restarted manually (by writing 1 to the appropriate START field in Main Configuration (MCR)) after exiting Power Down state.

NOTE

Writing 0 to MCR[PWDN] and writing 1 to MCR[NSTART] or MCR[JSTART] during the same cycle is not supported.

57.3.14 Low Power mode support

For low-power operation, ADC must be in Idle state and then put into Power Down state to ensure proper signal condition at the analog boundary.

From any of the states listed below, follow the sequence of steps shown to put ADC into Power Down state.

1. Idle condition
    a. Verify that ADC is in Idle state (MSR[ADCSTATUS] = 000b).
    b. Write 1 to MCR[PWDN], to shut off the clock to the hard macro with the proper state of signals required for Low Power mode.
    c. Verify that ADC is in Power Down state (MSR[ADCSTATUS] = 001b).
2. During calibration

    NOTE

    Writing 1 to MCR[PWDN] during calibration is prohibited. After calibration is started it should be allowed to finish normally. Alternatively, it can be terminated by starting a normal conversion before writing 1 to MCR[PWDN].

    a. Check the current ADC state either via MSR[ADCSTATUS] (000b = Idle, 011b = Calibration), or via CALBISTREG[C_T_BUSY] (1 = calibration is in progress).
    b. When ADC is in Idle state, write 1 to MCR[PWDN] to shut off the clock to the hard macro with the proper state of signals required for Low Power mode.
    c. Verify that ADC is in Power Down state (MSR[ADCSTATUS] = 001b).
3. During conversion (normal [one-shot]/Injected/BCTU):
    a. Write 1 to MCR[PWDN].
    b. ADC enters Power Down mode after completing the current conversion chain (for normal and injected conversions) or the conversion for conversion triggered by BCTU.
    c. Check the ADC state via MSR[ADCSTATUS] (000b = Idle state, 001b = Power Down state).
4. During Scan mode operation of normal conversion

    NOTE

    Writing 1 to MCR[PWDN] during scan mode operation has no effect other than to prevent software from starting a new normal or injected conversion. It does not stop a conversion in progress and cannot put ADC into Power Down state.

    a. Software must write 0 to MCR[NSTART] to stop the ongoing scan conversion before (or after) writing 1 to MCR[PWDN]. This stops conversion at the current chain boundary.
    b. ADC enters Power Down mode after coming to an Idle state after MCR[PWDN] is written with 1. This shuts the clock off to the hard macro and puts analog inputs in the proper state required for Low Power mode.
    c. Verify that ADC is in Power Down state (MSR[ADCSTATUS] = 001b).

57.3.15 Auto Clock Off mode

To reduce power consumption during Idle state (without going into Power Down state), an auto-clock-off feature can be enabled by writing 1 to MCR[ACKO]. When enabled, the internal ADC operating clock (AD_Clk) is automatically switched off during an idle period (for example, no conversion is programmed).

57.3.16 Calibration and self-test

57.3.16.1 Calibration

ADC must be calibrated before it can do meaningful conversions. The application must prevent any start of conversion before the ongoing calibration has finished successfully. No conversion trigger may be received during calibration.

During calibration, a known reference voltage is sampled and converted under controlled conditions to determine the correction values (calibration values) for offset, gain, and capacitor mismatch.

These calibration values (except gain calibration) are used in a post-processing step after conversion, reducing or eliminating the various error contribution effects. The gain calibration is used during sample phase to define the additional charge to be loaded to compensate for gain failure.

You must run calibration after every power-up reset (check device-specific reset connectivity for appropriate reset applied). You must also run calibration if it is indicated by the self-test (see Self-test).

Configuration and start of calibration is done by programming the Control And Calibration Status (CALBISTREG). During calibration, internal hardware averaging function should be enabled. Maximum averaging is recommended, and 16 should be the minimum.

Calibration status is captured by a status flag (CALBISTREG[TEST_FAIL]) to indicate any fault detected during calibration execution or if calibration was prematurely aborted. Calibration is aborted if any normal conversion is initiated during the calibration execution, which is called premature termination. (Flag TEST_FAIL = 1 indicates a calibration fail, for example when a calibration value is out of internally defined limits.)

To execute the calibration algorithm, perform the following steps:

1. Select the conversion clock frequency to be within the allowed limits for the calibration.
2. Bring ADC from Power Down state to active conversion (program MCR[PWDN] = 0b).
3. Configure the Control And Calibration Status (CALBISTREG). The default values are set for maximum accuracy (recommended).
4. Disable the error correction for the smaller capacitances in the CDAC of the ADC (write 0h to CAL2[ENX]).
5. Start calibration (program CALBISTREG[TEST_EN] = 1b), and calibration starts immediately.
6. Poll the status of CALBISTREG[C_T_BUSY] for 0 (wait until it transitions to 0).
7. Check CALBISTREG[TEST_FAIL] to determine the final status. If 1, then calibration failed.
8. Check the status of MSR[CALIBRTD]. If calibration was successful this field is 1.

57.3.16.2 Self-test

For safety applications, it is important to verify correct operation at regular intervals. ADC has a self-test feature for this purpose. When self-test is enabled, ADC automatically checks its components and flags errors, if any are found.

Tests can be enabled to check the reference (VrefH) and calibrated values.

The following test algorithms have been implemented:

- Supply self-test (algorithm S): It includes the conversion of the band gap and VREF voltages. It includes a sequence of num_supply_steps test conversions (steps). The supply test conversions must be an atomic operation (that is, all supply algorithm conversions must be performed one after another with no functional conversions in between).

- Capacitive self-test (algorithm C): It includes a sequence of test steps per definition of algorithm C (see Table 277) which executes the capacitive matrix of the CDAC used for sampling and conversion.

Individual steps can take up to 1 µs at 80 MHz ADC clock frequency.

Table 277. Self-test steps

STCR3[ALG]	STCR3 [step](per algorithm definition)	Description	Outcome	Comment
00 (algorithm S)	0	Supply self-test (band gap voltage)	Measures the internal band gap voltage	
	1	Supply self-test (reference voltage high)	Measures ADC high reference voltage	
	2	Supply self-test (reference voltage high)	Measures ADC high reference voltage	
01 (Reserved)	Reserved			
10 (algorithm C)	0 – 11	Capacitive self-test. One of the calibration steps is being run.	The difference/error of the individual offset value from the previously calibrated value is being returned as an ADC result that is compared with the programmed value in the self-test analog watchdog register(s) to detect any fault.	The ideal result value is expected to be 0. If the returned value is higher, it indicates a runtime fault or accuracy shift. The error is indicated in the status register and can be used to generate an interrupt also. It that indicates ADC should be recalibrated to check whether the reported error can be managed by calibration or that permanent damage occurred to ADC.
11 (algorithm S + C)	0 – 2 (S), 0 – 11 (C)	Supply for One-Shot mode and (supply + capacitive) for Scan mode only.	Same as above.	

ADC:

- Schedules self-testing algorithms using configuration registers.
- Monitors the converted data using analog watchdog registers.
- Flags any errors to a fault control unit (FCCU), if one exists.

Self-test steps can be activated from software (CPU) or BCTU.

ADC mode	TEST algorithm (CPU)	TEST algorithm/step (BCTU)
CPU mode (MCR[BCTUEN] = 0)	Yes	No

Table continues on the next page...

Table continued from the previous page...

ADC mode	TEST algorithm (CPU)	TEST algorithm/step (BCTU)
	• One-Shot operation mode • Scan operation mode	
BCTU mode	No	Yes

57.3.16.2.1 CPU mode

In this mode, self-test conversion is similar to normal conversion. You enable self-test by writing 1 to STCR2[EN].

Self-test conversions execute along with the functional conversions. The sequencing of steps of the selected algorithm depends on the operating mode of normal conversions (selected by MCR[MODE]).

In One-Shot operation mode, if self-test is enabled, only one step of the selected self-testing algorithm is executed at the end of the chain. The step number and algorithm to be executed are programmed in Self-Test Status 3 (STSR3). In One-Shot operation mode the sequence is:

1. Program NCMR*n* to select channels to be converted for normal conversion.
2. Write 0 to MCR[MODE] to select one-shot operation.
3. Configure the self-test algorithm threshold values (see Table 278).
4. Program sampling duration values in STCR1[INPSAMP*m*] (*m* = S, C).
5. Select the self-testing algorithm (in STCR3[ALG]), and step (in STCR3[MSTEP]). The default is algorithm S, step 0.
6. Enable self-testing by writing 1 to STCR2[EN].
7. Start the normal conversion by writing 1 to MCR[NSTART].
8. All normal conversions are executed as usual.
9. At the end of all normal conversions in the chain, the step number (programmed in STCR3[MSTEP]) of the self-testing algorithm (selected by STCR3[ALG]), is executed similar to a normal functional channel.
10. At the end of the conversion for the self-test channel:
    - The result is written to STDR1[TCDATA].
    - STDR1[VALID] transitions to 1.
    - ISR[EOC], ISR[ECH], and STSR1[ST_EOC] transition to 1.
11. ADC returns to Idle state.

For example: Channels A→B→C→D→E→F→G→H are present in the device where channels B→D→E are to be converted in One-Shot operation mode. At the end of conversion for channels B→D→E, self-test conversion is performed and ISR[ECH] and ISR[EOC] transition to 1. The sequence is B→D→E→self-test step.

MSR[NSTART] transitions to 1 when the normal conversion starts and transitions to 0 at the end of conversion for the test channel.

In Scan operation mode, consecutive steps of the selected self-test algorithm are converted continuously at the end of each chain of normal conversions. The number of channels converted at the end of each chain is 1 (except for algorithm S, in which all the steps are performed sequentially without any functional conversion interleaved). So, in Scan operation mode the sequence is:

1. Program NCMR*n* to select the channels to be converted for normal conversion.
2. Write 1 to MCR[MODE] to select Scan operation mode.
3. Configure the self-test algorithm threshold values (see Table 278).
4. Program sampling duration values in STCR1[INPSAMP*m*] field (*m* = S, C).

5. Select the self-testing algorithm in STCR3[ALG]. By default, all algorithms (supply and capacitive) are selected, that is, all algorithms are executed step-by-step, one after the other.
6. Enable self-testing by writing 1 to STCR2[EN].
7. Start the normal conversion by writing 1 to MCR[NSTART].
8. All normal conversions are executed as usual.
9. At the end of the chain of normal conversions (assuming the default value of STCR3[ALG]), all steps of algorithm S are performed (as algorithm S is always atomic). MSR[SELF_TEST_S] transitions to 1.
10. At the end of the conversion of the self-test channel for the last step of algorithm S, the result is written to STDR1[TCDATA] and STDR1[VALID] transitions to 1. At the same time, MSR[SELF_TEST_S] transitions to 0. The following registers also transition to 1:
    - ISR[EOC]
    - ISR[ECH]
    - STSR1[ST_EOC]
11. The next chain of normal conversions then starts.
12. At end of the chain of normal conversions, ADC executes step 0 of algorithm C.
13. At the end of the conversion of the test channel for step 0 of algorithm C, the result is stored in STDR1[TCDATA] and STDR1[VALID] transitions to 1 (if STSR1[OVERWR] is 1). Also, ISR[EOC] and ISR[ECH] transition to 1. STSR1[ST_EOC] also transitons to 1.
14. The next chain of normal conversion then starts.
15. At the end of the normal conversion chain, ADC executes step1 of algorithm C.

This process continues for all steps of all three algorithms. The state machine returns to Idle state when MCR[NSTART] transitions to 0.

For example: channels A→B→C→D→E→F→G→H are present in the device where channels B→D→E are to be converted in Scan operation mode. At the end of every conversion chain for channels B→D→E, one step of the self-test conversion is performed and ISR[ECH] and ISR[EOC] transition to 1.

The sequence is:

B→D→E→ST-S0→ST-S1→ST-S2→B→D→E→ST-C0→B→D→E→ST-C1→B→D→E→ST-C2→ . . . B→D→E→ST-C11→B→D→E→ST-S0 . . .

ST-Sx - Self-test supply step x (x = 0, 1, 2)

ST-Cx - Self-test capacitive step x (x = 0 to 11)

MSR[NSTART] transitions to 1 when normal conversion starts.

NOTE

- If instead of starting normal conversion by software (by writing 1 to MCR[NSTART]), it is started by an external trigger, the self-test behavior remains the same.
- Self-test channel conversion is not performed for injected conversions. It is performed only during normal conversions.
- If, during a test channel conversion, an injected conversion arrives, the test conversion is aborted (just as a normal functional channel) and the injected conversion is performed. After the injected conversion is complete, the test conversion resumes from the step at which it was aborted. In this case, MSR[SELF_TEST_S] remains 1 during the injected conversion.
- For self-test, MCR[MODE] should be programmed (at least one cycle) before writing 1 MCR[NSTART] and should not be changed until the conversion is finished or terminated.

Table 278. Recommended self-test threshold values

Register field	Hex value	Decimal value	Comment
STAW0R[THRH]	3669h	13929	The threshold values depend on the stability of the supply and the reference voltage. When you see voltage variations, widen the limits.
STAW0R[THRL]	5 V reference: 1A7Fh 3 V reference: 2B27h	5 V reference: 6784 3 V reference: 11047	
STAW1R[THRL]	3FF9h	16377	
STAW2R[THRL]	3FF9h	16377	
STAW4R[THRH]	20h	32	
STAW4R[THRL]	7FE0h	–32	
STAW5R[THRH]	20h	32	
STAW5R[THRL]	7FE0h	–32	

57.3.16.2.2 BCTU mode

BCTU mode is enabled by writing 1 to MCR[BCTUEN].

Self-test conversion behaves in the same way for all BCTU operating modes (Trigger mode and Control mode). In both cases, if MCR[BCTUEN] = 1, the self-test conversion can be started only by BCTU—software cannot start it. MCR[BCTUEN] must not be changed during normal conversion. The interface between BCTU and ADC (for self-test) is shown in the following figure.

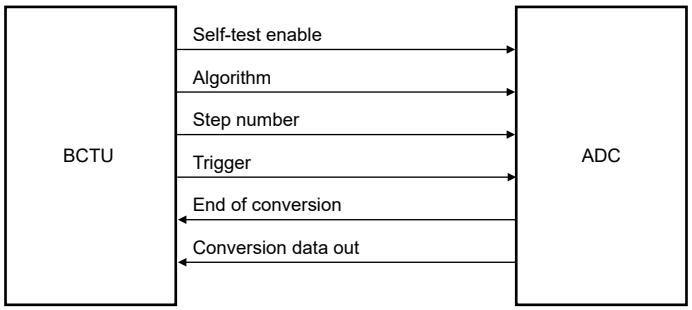


Figure 233. Self-test interface between ADC and BCTU

For self-test conversions in BCTU mode, BCTU asserts self-test enable along with the Trigger command. The algorithm and the step number to be executed is put on algorithm and step number, respectively. The other commands (Trigger, End Of Conversion, and Conversion Data Out) have the same meaning, like normal BCTU functional conversion.

As already mentioned for algorithm S, the three steps must be atomic. In BCTU mode, this is managed by BCTU, that is, BCTU sends three triggers (one for each step) asserting self-test enable and updating test_step for each step.

57.3.16.2.3 Abort and abort chain for self-testing channel

Writing 1 to MCR[ABORT] during self-test channel conversion has no effect.

In One-Shot operation mode, if you write 1 to MCR[ABORTCHAIN] during the self-test channel conversion, the self-test channel is aborted and ISR[ECH] transitions to 1. In this case, ISR[EOC] for the self-test channel is not generated.

In Scan operation mode, if MCR[ABORTCHAIN] = 1 when self-test channel step *n* is ongoing, self-test channel step *n* is aborted and the next chain conversion begins. At the end of the chain, the step *n* conversion is performed again. (For algorithm S, the full algorithm is executed again).

57.3.16.2.4 Self-test analog watchdog

ADC provides monitoring options for conversion data generated by self-test algorithms. Analog watchdogs determine whether the conversion results for self-test algorithms are in the range of a particular guard area. ADC provides separate self-test analog watchdog registers for each algorithm.

After the conversion of each step of a self-test algorithm, ADC compares the converted value and the threshold values if the analog watchdog feature has been enabled by writing 1 to STAW*n*R[AWDE] (for example, STAW0R[AWDE]). If the converted value is not between the upper and lower threshold values specified by the Self-Test Analog Watchdog Register for the particular algorithm, the corresponding error flag, STSR1[ERR_x], transitions to 1 and the step number in which error occurred is captured in STSR1[STEP_x] (for algorithm C). Erroneous data is written to STSR4[DATA*n*]. The STSR1[ERR_x] flags generate an interrupt if enabled by the corresponding mask bit in Self-Test Configuration 2 (STCR2). The fault indication is also forwarded to the fault control unit, if present, so that necessary action can be taken at the chip level. The fault type (critical or noncritical) is specified by the configuration in Self-Test Configuration 2 (STCR2).

The analog watchdog feature works differently for algorithm S. Algorithm S is always an atomic operation. Self-Test Status 1 (STSR1) has a separate error field for each step of algorithm S to avoid overwrite in case an error occurs in more than one step. Therefore, there are separate mask bits for each step in STCR1. For the same reason, the status registers (Self-Test Status 2 (STSR2) and Self-Test Status 3 (STSR3)) have separate fields for each step to store erroneous data.

In step 0 of the supply algorithm, ADC measures the band gap voltage (1.2 V), which is assumed to be stable. The conversion result of step 0 is compared against high (THRH) and low (THRL) thresholds as defined in Self-Test Analog Watchdog S0 (STAW0R), if enabled (STAW0R[AWDE] = 1). The STSR1[ERR_S0] flag transitions to 1 if any of the thresholds are violated. STSR1[ERR_S0] is cleared by writing 1 to it. The conversion result of ADC can be calculated with the following formula:

- ADC data (decimal) = (Vin × 2^{15} ÷ (Vrh - Vrl))

Where:

- Vin: input voltage (band gap voltage in this case)
- Vrh: ADC reference high voltage
- Vrl: ADC reference low voltage

The upper and lower threshold value can be calculated with the above formula using maximum and minimum values of Vin, Vrh, Vrl (Vin is the band gap supply voltage in this case):

- Upper threshold (THRH, decimal) = (Vin[max] × 2^{15}) ÷ (Vrh[min] - Vrl[max])
- Lower threshold (THRL, decimal) = (Vin[min] × 2^{15}) ÷ (Vrh[max] - Vrl[min])

The minimum and maximum voltages can be obtained from the chip data sheet. The band gap voltage tolerance, as well as the actual reference voltage used and its tolerances, must be considered when calculating high and low thresholds.

For algorithm S steps 1 and 2, (VREF/VREF) is measured in order to check the integrity of the sampling signal. For these particular conversions, no higher threshold value is required as the ideal value is FFFh. Only the lower threshold value is programmed in Self-Test Analog Watchdog S1 (STAW1R) and Self-Test Analog Watchdog S2 (STAW2R).

For algorithm C, a separate register is provided for step 0. In step 0 an offset for other steps is measured. The converted data is compared with the threshold values in Self-Test Analog Watchdog C (STAW5R) if STAW4R[AWDE] = 1. For other steps, this offset is subtracted from converted data before performing watchdog checks.

57.3.16.2.5 Watchdog timer

The watchdog timer is an additional check that monitors the sequence of the self-testing algorithm that has been implemented, and also checks whether the algorithm is completed within a safe time period. The watchdog timers are enabled for CPU as well as BCTU conversions. Each algorithm has an independent watchdog timer. The watchdog timer for a particular algorithm is enabled by writing 1 to STAWxR[WDTE]. The safe time value is programmed in STBRR[WDT] (the default value is 10 ms, assuming an 80 MHz clock).

The safe time is measured starting from step 0 of the algorithm (including all normal chain conversions in between) to the point where step 0 of the same algorithm starts again.

The programming sequence is:

1. Program NCMR*n* to select the channels for normal conversion in Scan operation mode (MCR[MODE] = 1).
2. Select the self-test algorithm in STCR3[ALG]. By default, all algorithms (supply and capacitive) are selected (all algorithms are executed step-by-step, one after the other).
3. Enable the self-test channel by writing 1 to STCR2[EN].
4. Program the safe period value in STBRR[WDT].
5. Enable the watchdog timer by writing 1 to STAWxR[WDTE]. Assume writing 1 to STAWxR[WDTE] occurs at time *t*0. It is important to do all configuration programming before writing 1 to STAWxR[WDTE], because the safe time check is also performed between writing 1 to STAWxR[WDTE] and the start of step 0. This is to verify that the algorithm has started within the safe time.
6. Start the normal conversion by writing 1 to MCR[NSTART].
7. At the end of the first conversion chain, three steps (steps 0, 1, and 2) of algorithm S are executed in sequence. Assume the start of Step 0 occurs at time *t*1.
8. After completion of algorithm S, ADC performs conversion of the next chain.
9. At the end of the the conversion chain, the first step (step 0) of algorithm C is performed.
10. In this way, one step of algorithm C is performed after completion of one conversion chain until all steps are finished. Steps are executed in sequence (steps 0–11)
11. After the last step of algorithm C, another chain conversion is executed. At the end of this chain conversion, step 0 of algorithm S is started again, repeating the whole sequence. Assume this time (starting of step 0 of the supply algorithm) to be at time *t*2.
12. For algorithm S, if (*t*1 - *t*0) > safe period or (*t*2 - *t*1) > safe period, the watchdog timer flags an error and STSR1[WDTERR] transitions to 1. ADC asserts a fault signal per the criticality configuration (critical or noncritical) and an interrupt is generated, if enabled (STCR2[MSKWDTERR] = 1). Otherwise, the watchdog timer counter is reset and starts again to monitor the next sequence.
13. A similar sequence is followed for watchdog timers for algorithm C.

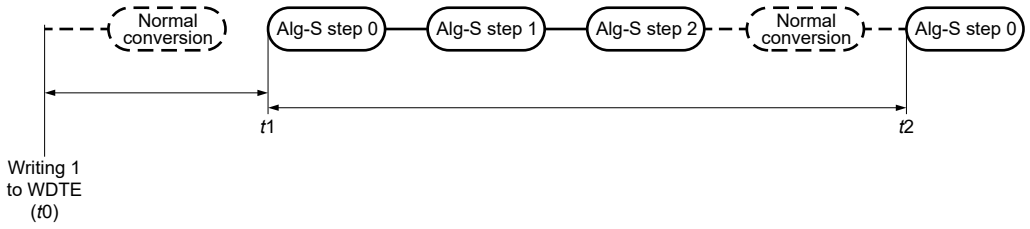


Figure 234. Watchdog timer monitor for algorithm S

NOTE

1. Because BCTU may not incorporate any safe period checking mechanism, the watchdog timers can also be enabled for BCTU conversions.
2. You must disable watchdog timers for self-test algorithms that are not executed.

57.3.16.2.5.1 Watchdog sequence checking

The self-test watchdog timer incorporates sequence checking features to verify that the steps of each algorithm are executed in the correct order. If the steps are not in the correct order, STSR1[WDSERR] transitions to 1 to indicate the error. A fault is indicated by an interrupt flag if STCR2[MSKWDSERR] = 1. A fault is also asserted as per criticality configuration (critical or noncritical) and an interrupt generated if STCR2[MSKWDSERR] = 1.

A watchdog sequence error is flagged in the following cases:

- The steps of any algorithm are not executed in proper order.
- The step numbers provided by BCTU for a BCTU conversion are not in order. Watchdog sequence checking is significant only for BCTU burst mode.
- When an abort chain occurs during a self-test channel conversion, that step must be repeated at the end of the next chain. This generates a sequence error as soon as the self-test channel conversion resumes. There is an exception, however. If an abort chain occurs during the last step of algorithm S, the sequence error is not flagged because the whole algorithm must be repeated.

If an injected conversion occurs during self-test channel conversion, a watchdog sequence error is not flagged, although the ongoing step number is aborted and is repeated.

NOTE

The watchdog timer feature is applicable for Scan operation mode but not for One-Shot operation mode.

57.3.16.2.6 Baud rate control for test channel

The baud rate control feature controls the scheduling of self-test channel conversions between normal conversion chains. The scheduling rate is specified by STBRR[BR].

By default, if the self-test channel is enabled, one step of the selected algorithm is executed after each normal conversion chain. The bandwidth consumed by the self-test channel depends on the number of channels in the normal chain. For example, if you have 50 normal conversions in a chain, then the self-test channel consumes only 2% of the total bandwidth, which is very small. If the number decreases to just 3 channels, the bandwidth consumed by the self-test channel is 25%, which is significant (and may not be desirable because it slows down the normal conversion rate).

STBRR[BR] provides flexibility by scheduling the self-test channel conversion to be performed not at the end of every chain, but at the end of BR + 1 chains. For example, if STBRR[BR] = 5, a single step of the selected algorithm for the test channel is performed after 6 chain conversions. The next step is performed at the end of the next six chain conversions, and so on. By default, STBRR[BR] = 0.

NOTE

This feature is applicable only for Scan operation mode and not for One-Shot operation mode. STBRR[BR] should be written with 0 for One-Shot operation mode.

To use the baud rate control feature in Scan operation mode, NCMR*n* should have a nonzero value.

57.3.16.2.6.1 Abort chain when baud rate is nonzero

As already described, for a nonzero value of STBRR[BR], the self-test channel conversion is performed at the end of BR + 1 chains. If an abort chain occurs during the chain in which a self-test channel conversion is scheduled to be performed, the self-test channel conversion is performed after the next BR + 1 chains.

For example, if STBRR[BR] = 2, the sequence is two normal chains (without any self-test channel conversion), followed by a chain with the self-test channel converted at the end. If an abort chain occurs during the first two chains, it is treated as a normal chain abort and the self-test channel is converted at the end of the third chain only (as in the case without any abort chain). However, if an abort chain occurs during the third chain in which the self-test channel is scheduled to be converted, the self-test channel is converted after the next three chains (for example, at the end of the sixth chain, counting from the beginning).

57.3.17 Conversion time

Total conversion time depends on the conversion clock frequency, which is configured by programming MCR[ADCLKSEL].

The components of conversion time and affecting configurations are:

- PST
- ST
- CT

- DP
- TPT

Presample phase time is equal to the sample time with a one AD_clk cycle delay for phase transition from presample phase to sample phase. CTR*n*[INPSAMP] controls sample time for different types of channels and is specified in terms of AD_clk cycles, so the presampling time is sample time plus one cycle of AD_clk. The minimum value of sample time is 8. If the value programmed is less than 8, it has no effect on sample time and the sample time is 8 AD_clk cycles.

Compare phase time is controlled by:

- The evaluation time of a single bit
- The number of bits converted

For *n*-bit conversions, the value of CT is *n* multiplied by the evaluation time of a single bit in terms of AD_clk cycles.

The evaluation time of a single bit is 4 AD_clk cycles if AMSIO[HS] = 0h, and 5 AD_clk cycles if AMSIO[HS] = 1h.

The data processing time is 2 cycles of AD_clk. During these cycles raw converted data is corrected for offset, gain, capacitor mismatch, and so on.

Trigger processing time consists of:

- One module_clock cycle, which is required to prepare the channel and calculate the initial gain value used by ADC for the first conversion.
- BCTU triggering time:
  - Triggers from the synchronous BCTU interface require 1 cycle of the module clock for processing. One more cycle is required to register the BCTU trigger in synchronous mode.

The total conversion time, in terms of the module clock cycles, is calculated using the following equation for normal and injected conversions:

Total_conversion_time = ([(PST + ST + CT + DP) × chain_length] + TPT) × TAD_clk

Example:

The ADC controller clock is equal to the module clock (80 MHz, clock cyle = 12.5 ns)

- ADC resolution 12 bit + 1 bit for a special capacitor (CS)
- Three channels are programmed in NCMR*n*, so a chain of three is to be converted
- Default sample time (22 cycles) is specified
- No presampling
- Conversion time (4 cycles per bit)

The total time for the three conversions = [(0 + 22 + (4×13) + 2) × 3] + 1 = 229 cycles ~= 2.862 µs

Conversion timing in Calibration for full test:

- Single conversion time = [sample time (CALBISTREG[TSAMP]) + compare time (11 × 4) + data proc(2)]
- Inter-conversion gap = 1 cycle
- Total number of tests = 12
- Averaging = 32

One test duration = (single conversion time + inter-conversion gap) × averaging samples

Total test duration = ((one test duration) × total number of tests).

Example: sample time = 22

Total test time = ((((22 + (11×4) +2)+1) × 32) × 12) = 26496 cycles × 12.5 ns (80 MHz) = 331200 ns

Conversion time in self-test:

Self-test conversion time equals one test period multiplied by the number of consecutive (atomic) steps. For supply self-test, the atomic step is 3 and for other tests it is 1.

One test period for self-test is as below.

- For supply tests: ([sample_time + (13 × 4) + 2])
- For other algorithms: ([sample_time + (11 × 4) + 2])

NOTE

Algorithm S steps (conversions) always run in atomic operation, required for an algorithmic procedure. In Scan operation mode, when all algorithms are selected, three steps of algorithm S run together after the end of the conversion chain. algorithm C steps are not required to be run together, so after each conversion chain one step is executed serially. Conversion time with self-test must also be calculated accordingly.

Example.

- Channels in chain A→B→C.
- Scan operation mode.
- All self-test algorithms are selected.

So, the resulting conversion order is:

[A→B→C→AS_step0→AS_step1→AS_step2→A→B→C→AC_step0→A→B→C→AC_step1→A→B→C→AC_step2→.....→A→B→C→AC_step11]→[same sequence repeats]

NOTE

- AS = Algorithm S (supply self-test)
- AC = Algorithm C (capacitive self-test)

57.3.18 Conversion data processing

Raw converted data contains many types of errors, such as offset, gain, and so on. ADC processes raw conversion data before writing the result to a data register, to reduce or eliminate error contributions. The process of error correction happens in parallel with bit-by-bit evaluation, using the correction values generated during the offset determination and calibration process.

The error correction value is subtracted from the raw conversion result whenever the comparator output for one of the calibrated capacitors evaluates high during the conversion period.

To reduce data processing time, error correction is done in parallel during each bit evaluation of a conversion. Lower-weighted capacitors (smaller than C9) are not calibrated, so no error correction is performed for these capacitors. However, they are self-tested to guarantee that the error contribution is below specified accuracy limits.

The final result is checked for any overflow or underflow. When the processed data is above the maximum value that can be represented by ADC resolution, the output data is forced to all 1s (FFFh for 12-bit resolution). Similarly, if the processed data is negative then output data is forced to all 0s (000h for 12-bit resolution).

57.3.19 User-defined offset and gain values

In addition to the ADC-calculated offset value and gain value, ADC enables you to specify another set of offset and gain values in Offset And Gain User (OFSGNUSR). These values must be stored in two's complement format, where the leftmost bit (the most significant bit) is the sign bit.

ADC subtracts these values from the final processed conversion result. The values are used to removed fixed DC bias or any other known errors—for instance those caused by board design, sensors, and so on.

57.4 Clock frequency

ADC's internal blocks are designed for an input clock frequency up to the Fin (Input Signal Bandwidth) maximum (see the ADC electrical specification section in the chip data sheet). The specified speed can be achieved with this clock frequency. If the

input clock frequency exceeds the specified maximum frequency, ADC accuracy degrades and, in the worst case scenario, ADC malfunctions. ADC accuracy also degrades if the input clock frequency is below the specified minimum frequency. Sampling time has an absolute maximum value to guarantee the specified parameters and accuracy. For slower clocks, the sampling clock count must be programmed accordingly.

57.5 Memory map and register definition

ADC has a wide range of configurations to tailor its operation to application requirements. This comes with the drawback of possible corner cases. To avoid unintended behavior, configure ADC before starting a conversion and change any configuration only when ADC is idle, in other words when MSR[ADCSTATUS] = 0.

57.5.1 Transfer error description

The following register accesses cause a transfer error and do not change register content:

- Any access to an unused address.
- Any write access to a read-only register.

57.5.2 ADC register descriptions

57.5.2.1 ADC memory map

ADC_0 base address: 400A_0000h

ADC_1 base address: 400A_4000h

ADC_2 base address: 400A_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Main Configuration (MCR)	32	RW	0000_0001h
4h	Main Status (MSR)	32	R	0000_0001h
10h	Interrupt Status (ISR)	32	RW	0000_0000h
14h	Channel End Of Conversion Flag For Precision Inputs (CEOCFR0)	32	RW	0000_0000h
18h	Channel End Of Conversion Flag For Standard Inputs (CEOCFR1)	32	RW	0000_0000h
1Ch	Channel End Of Conversion Flag For External Inputs (CEOCFR2)	32	RW	0000_0000h
20h	Interrupt Mask (IMR)	32	RW	0000_0000h
24h	EOC Interrupt Enable For Precision Inputs (CIMR0)	32	RW	0000_0000h
28h	EOC Interrupt Enable For Standard Inputs (CIMR1)	32	RW	0000_0000h
2Ch	EOC Interrupt Enable For External Inputs (CIMR2)	32	RW	0000_0000h
30h	Analog Watchdog Threshold Interrupt Status (WTISR)	32	RW	0000_0000h
34h	Analog Watchdog Threshold Interrupt Enable (WTIMR)	32	RW	0000_0000h
40h	Direct Memory Access Configuration (DMAE)	32	RW	0000_0000h
44h	DMA Request Enable For Precision Inputs (DMAR0)	32	RW	0000_0000h
48h	DMA Request Enable For Standard Inputs (DMAR1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4Ch	DMA Request Enable For External Inputs (DMAR2)	32	RW	0000_0000h
60h - 6Ch	Analog Watchdog Threshold Values (THRHLR0 - THRHLR3)	32	RW	7FFF_0000h
80h	Presampling Control (PSCR)	32	RW	0000_0000h
84h	Presampling Enable For Precision Inputs (PSR0)	32	RW	0000_0000h
88h	Presampling Enable For Standard Inputs (PSR1)	32	RW	0000_0000h
8Ch	Presampling Enable For External Inputs (PSR2)	32	RW	0000_0000h
94h	Conversion Timing For Precision Inputs (CTR0)	32	RW	0000_0016h
98h	Conversion Timing For Standard Inputs (CTR1)	32	RW	0000_0016h
9Ch	Conversion Timing For External Inputs (CTR2)	32	RW	0000_0016h
A4h	Normal Conversion Enable For Precision Inputs (NCMR0)	32	RW	0000_0000h
A8h	Normal Conversion Enable For Standard Inputs (NCMR1)	32	RW	0000_0000h
ACh	Normal Conversion Enable For External Inputs (NCMR2)	32	RW	0000_0000h
B4h	Injected Conversion Enable For Precision Inputs (JCMR0)	32	RW	0000_0000h
B8h	Injected Conversion Enable For Standard Inputs (JCMR1)	32	RW	0000_0000h
BCh	Injected Conversion Enable For External Inputs (JCMR2)	32	RW	0000_0000h
C4h	Delay Start Of Data Conversion (DSDR)	32	RW	0000_0000h
C8h	Power Down Exit Delay (PDEDR)	32	RW	0000_0000h
100h - 11Ch	Precision Input n Conversion Data (PCDR0 - PCDR7)	32	R	0000_0000h
180h - 1DCh	Standard Input n Conversion Data (ICDR0 - ICDR23)	32	R	0000_0000h
200h - 27Ch	External Input n Conversion Data (ECDR0 - ECDR31)	32	R	0000_0000h
2B0h	Channel Analog Watchdog Select For Precision Inputs (CWSELRPI0)	32	RW	0000_0000h
2B4h	Channel Analog Watchdog Select For Precision Inputs (CWSELRPI1)	32	R	0000_0000h
2C0h	Channel Analog Watchdog Select For Standard Inputs (CWSELRSI0)	32	RW	0000_0000h
2C4h	Channel Analog Watchdog Select For Standard Inputs (CWSELRSI1)	32	RW	0000_0000h
2C8h	Channel Analog Watchdog Select For Standard Inputs (CWSELRSI2)	32	RW	0000_0000h
2D0h	Channel Analog Watchdog Select For External inputs (CWSELREI0)	32	RW	0000_0000h
2D4h	Channel Analog Watchdog Select For External inputs (CWSELREI1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2D8h	Channel Analog Watchdog Select For External inputs (CWSELREI2)	32	RW	0000_0000h
2DCh	Channel Analog Watchdog Select For External inputs (CWSELREI3)	32	RW	0000_0000h
2E0h	Channel Watchdog Enable For Precision Inputs (CWENR0)	32	RW	0000_0000h
2E4h	Channel Watchdog Enable For Standard Inputs (CWENR1)	32	RW	0000_0000h
2E8h	Channel Watchdog Enable For External Inputs (CWENR2)	32	RW	0000_0000h
2F0h	Analog Watchdog Out Of Range For Precision Inputs (AWORR0)	32	RW	0000_0000h
2F4h	Analog Watchdog Out Of Range For Standard Inputs (AWORR1)	32	RW	0000_0000h
2F8h	Analog Watchdog Out Of Range For External Inputs (AWORR2)	32	RW	0000_0000h
340h	Self-Test Configuration 1 (STCR1)	32	RW	1818_2507h
344h	Self-Test Configuration 2 (STCR2)	32	RW	0000_0005h
348h	Self-Test Configuration 3 (STCR3)	32	RW	0000_0300h
34Ch	Self-Test Baud Rate (STBRR)	32	RW	0005_0000h
350h	Self-Test Status 1 (STSR1)	32	RW	0000_0000h
354h	Self-Test Status 2 (STSR2)	32	R	0000_0000h
358h	Self-Test Status 3 (STSR3)	32	R	0000_0000h
35Ch	Self-Test Status 4 (STSR4)	32	R	0000_0000h
370h	Self-Test Conversion Data 1 (STDR1)	32	R	0000_0000h
380h	Self-Test Analog Watchdog S0 (STAW0R)	32	RW	0727_04C5h
388h	Self-Test Analog Watchdog S1 (STAW1R)	32	RW	0000_3FF9h
38Ch	Self-Test Analog Watchdog S2 (STAW2R)	32	RW	0000_3FF9h
394h	Self-Test Analog Watchdog C0 (STAW4R)	32	RW	0010_3FF0h
398h	Self-Test Analog Watchdog C (STAW5R)	32	RW	0010_3FF0h
39Ch	Analog Miscellaneous In/Out register (AMSIO)	32	RW	0000_0811h
3A0h	Control And Calibration Status (CALBISTREG)	32	RW	See section
3A8h	Offset And Gain User (OFSGNUSR)	32	RW	0004_0000h
3B4h	Calibration Value 2 (CAL2)	32	RW	4300_8243h

57.5.2.2 Main Configuration (MCR)

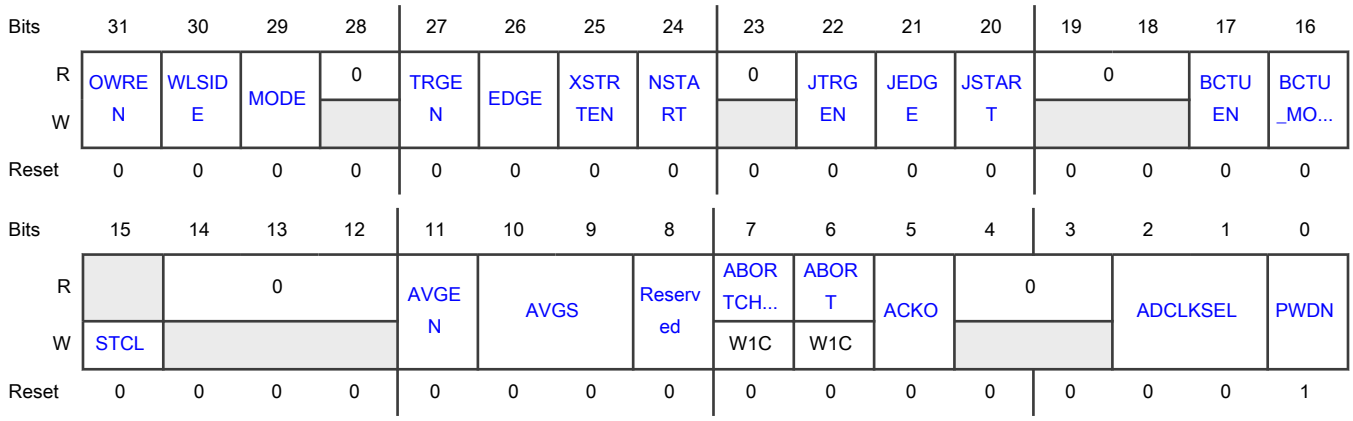
Offset

Register	Offset
MCR	0h

Function

Configures most ADC features. You must change field values only when ADC is in Idle state, except for the ABORTCHAIN and ABORT fields.

Diagram



Fields

Field	Function
31 OWREN	Overwrite Enable Specifies whether a conversion data register accepts new data before the current conversion data has been read. When enabled, the new conversion result overwrites the older data, regardless of the validity of the older conversion data. When PCDR*n*[VALID] = 1, the conversion data has not been read. 0b - Disable 1b - Enable
30 WLSIDE	Write Left-Aligned Specifies whether conversion data is right-aligned or left-aligned when written to one of the conversion data registers: • PCDR*n* • ICDR0*n* • ECDR*n* Right-aligned data occupies *CDR[14:0]. Left-aligned data occupies *CDR[15:1], 0b - Right aligned 1b - Left-aligned
29 MODE	Normal Conversion Mode Specifies whether the set of input channels selected for a normal conversion are converted only once per start event or continuously after a start event. If a conversion is done only once, ADC enters an idle state after conversion is complete. For continuous conversion, all input channels selected for the normal conversion are converted continuously in a loop. Stop the loop by writing 0 to NSTART.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Single conversion 1b - Continuous conversion
28 —	Reserved
27 TRGEN	External Trigger Enable Specifies whether the normal trigger input starts a conversion. 0b - Normal trigger input does not start a conversion 1b - Normal trigger input starts a conversion
26 EDGE	External Trigger Edge Selection Selects which edge of the normal trigger input starts a conversion. 0b - Falling edge 1b - Rising edge
25 XSTRTEN	Auxiliary External Start Enable Enables the auxiliary normal trigger source to start a conversion. You can use this field to synchronize the start of a conversion of two ADC instances. 0b - Disable 1b - Enable
24 NSTART	Start Normal Conversion Starts a normal conversion. If the continuous mode is selected (MODE = 1), the value of this field remains 1. Write 0 to this field to stop the in-progress conversion loop after conversion of the last input channel of the selected set of input channels is complete. If continuous mode is not selected (MODE = 0), this field automatically resets to 0 after writing 1. 0b - No effect 1b - Starts conversion
23 —	Reserved
22 JTRGEN	Injection Trigger Enable Enables the injected trigger input as a source to start a conversion. 0b - Disable 1b - Enable
21 JEDGE	Injected Trigger Edge Selection Selects which edge of the injected trigger input starts an injected conversion.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Falling edge 1b - Rising edge
20 JSTART	Injected Start Interrupts any ongoing normal conversion and starts an injected conversion. This field automatically resets to 0. If an injected conversion is already ongoing, the field value remains 1 until the next injected conversion starts. This field can only be written with 1. 0b - Injected conversion can be started 1b - Starts an injected conversion
19-18 —	Reserved
17 BCTUEN	Body Cross Trigger Unit Enable Enables BCTU as a trigger source. 0b - Disable 1b - Enable
16 BCTU_MODE	Body Cross Trigger Unit Mode Select Specifies whether sources other than BCTU can start a conversion when the BCTU is enabled as a trigger source (BCTUEN = 1). This field is writeable only when ADC is in Power Down state (PWDN = 1). 0b - Only BCTU can trigger conversion 1b - All trigger sources can trigger conversion
15 STCL	Self-Test Configuration Lock Protects the following registers from writing: • STCR1 • STCR2 • STCR3 • STBRR • STAW0R • STAW1R • STAW2R • STAW4R • STAW5R 0b - Registers are writeable 1b - Registers are read-only
14-12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11 AVGEN	Averaging Enable Enables conversion averaging. 0b - Disable 1b - Enable
10-9 AVGS	Averaging Select Specifies the number of conversions ADC uses to calculate the conversion result. 00b - 4 conversions 01b - 8 conversions 10b - 16 conversions 11b - 32 conversions
8 —	Reserved
7 ABORTCHAIN	Abort Chain Aborts the conversion of the selected set of input channels (chain of input channels). The currently ongoing conversion completes. This field always reads 0. This field cannot be programmed during BCTU conversion. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Undefined When writing 0b - Conversion continues 1b - Conversion aborted
6 ABORT	Abort Conversion Aborts an ongoing conversion. This field always reads 0 and cannot be written during BCTU conversion. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Undefined When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Conversion continues 1b - Conversion aborted
5 ACKO	Auto Clock Off Reduces power consumption by turning off the clock of the analog part of ADC when it is in Idle state. 0b - Clock always active 1b - Clock gated
4-3 —	Reserved
2-1 ADCLKSEL	Conversion Clock (AD_clk) Frequency Selection Selects the frequency for the clock signal of the conversion circuit (AD_clk = module clock ÷ ($2^{ADCLKSEL}$)). This field can be written only when ADC is in Power Down state (PWDN = 1). 00b - Module clock frequency 01b - Module clock frequency ÷ 2 10b - Module clock frequency ÷ 4 11b - Module clock frequency ÷ 8
0 PWDN	Power Down Reduces power consumption by turning off power to the analog portion of ADC. 0b - ADC enters a functional state 1b - ADC enters Power Down state

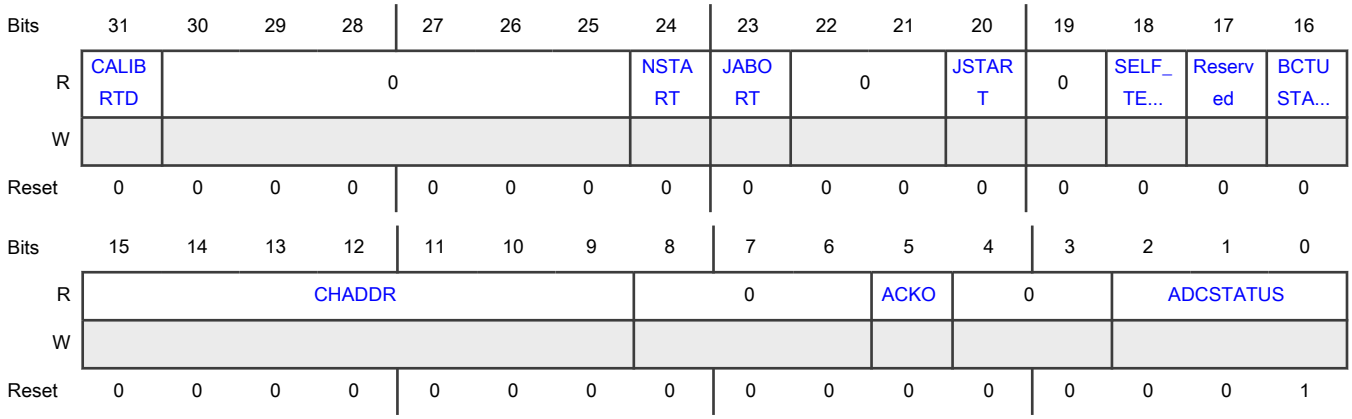
57.5.2.3 Main Status (MSR)

Offset

Register	Offset
MSR	4h

Function
Contains flags that indicate the current ADC state.

Diagram



Fields

Field	Function
31 CALIBRTD	Calibration Status Indicates ADC calibration status. 0b - Uncalibrated or calibration unsuccessful 1b - Calibrated
30-25 —	Reserved
24 NSTART	Normal Conversion Started Shows whether a normal conversion is in progress. 0b - Not in progress 1b - In progress
23 JABORT	Injected Conversion Aborted Indicates whether the conversion of a set of inputs selected for injected conversion has been aborted. This field resets to 0 when a new injected conversion is started. 0b - Not aborted 1b - Aborted
22-21 —	Reserved
20 JSTART	Injected Conversion Started Indicates whether an ongoing conversion was started by the injection trigger. 0b - Not an injected conversion 1b - Injected conversion

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 —	Reserved
18 SELF_TEST_S	Indicates whether an ongoing conversion is for self-test. 0b - Not self-test 1b - Self-test
17 —	Reserved
16 BCTUSTART	BCTU Conversion Started Indicates whether a BCTU conversion is ongoing. This field is 1 when a BCTU trigger event is received and the BCTU conversion starts. When the BCTU trigger mode is selected, this field is automatically reset to 0 when conversion is completed. Otherwise, if BCTU control mode is selected, this field resets to 0 when the BCTU is disabled (BCTUEN = 0). 0b - Conversion was not triggered by BCTU 1b - Ongoing conversion was triggered by BCTU
15-9 CHADDR	Input Under Measure Contains the number of the input that is currently converted. 000_0000b-011_1111b - Input number
8-6 —	Reserved
5 ACKO	Auto Clock-Off On Indicates whether the auto clock-off feature is active, and the conversion circuit is not receiving a clock signal. When auto clock-off is inactive, the conversion circuit is clocked. 0b - Inactive 1b - Active
4-3 —	Reserved
2-0 ADCSTATUS	ADC State Indicates the current state of the ADC Finite State Machine (FSM). 000b - Idle 001b - Power Down 010b - Wait

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - Calibrate 100b - Convert 110b - Done

57.5.2.4 Interrupt Status (ISR)

Offset

Register	Offset
ISR	10h

Function

Contains flags that indicate whether an interrupt has been generated.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											EOBCTU	JEOC	JECH	EOC	ECH
W												W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5 —	Reserved
4 EOBCTU	End Of BCTU Conversion Indicates the status of the End of BCTU Conversion (EOBCTU) interrupt, which is generated after a BCTU conversion completes. **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0b - No EOBCTU interrupt generated 1b - EOBCTU interrupt generated When writing 0b - No effect 1b - Clears flag
3 JEOC	End Of Injected Conversion Indicates the status of the End of Injected Conversion (JEOC) interrupt, which is generated after an injected conversion completes. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No JEOC interrupt generated 1b - JEOC interrupt generated When writing 0b - No effect 1b - Clears flag
2 JECH	End Of Injected Chain Conversion Indicates the status of the End of Injected Chain (JECH) interrupt, which is generated after injected conversion of a set of inputs completes. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No JECH interrupt generated 1b - JECH interrupt generated When writing 0b - No effect 1b - Clears flag
1 EOC	End Of Conversion Indicates the status of the End of Conversion (EOC) interrupt, which is generated after a normal conversion completes. **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0b - No EOC interrupt generated 1b - Interrupt generated When writing 0b - No effect 1b - Clears flag
0 ECH	End Of Chain Conversion Indicates the status of the End of Chain (ECH) conversion interrupt, which is generated after a set of inputs has been converted normally. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Indicates no ECH interrupt generated 1b - Indicates an ECH interrupt has been generated When writing 0b - No effect 1b - Clears flag

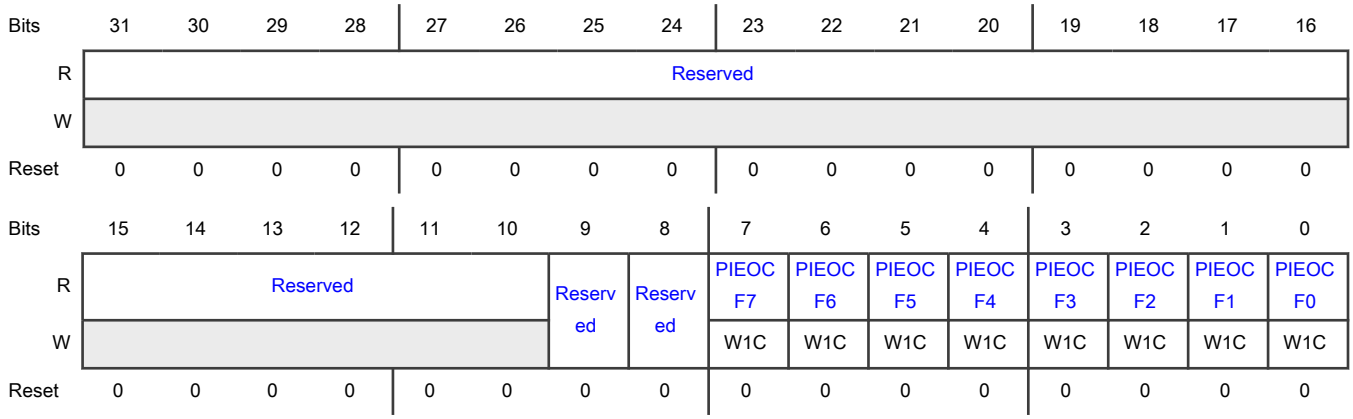
57.5.2.5 Channel End Of Conversion Flag For Precision Inputs (CEOCFR0)

Offset

Register	Offset
CEOCFR0	14h

Function
Contains flags that indicate whether conversion is complete for a precision input.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 PIEOCF7	Precision Input End Of Conversion Flag 7 Indicates whether conversion of a precision input 7 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
6 PIEOCF6	Precision Input End Of Conversion Flag 6 Indicates whether conversion of a precision input 6 has completed. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
5 PIEOCF5	Precision Input End Of Conversion Flag 5 Indicates whether conversion of a precision input 5 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
4 PIEOCF4	Precision Input End Of Conversion Flag 4 Indicates whether conversion of a precision input 4 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
3 PIEOCF3	Precision Input End Of Conversion Flag 3 Indicates whether conversion of a precision input 3 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Conversion complete When writing 0b - No effect 1b - Clears flag
2 PIEOCF2	Precision Input End Of Conversion Flag 2 Indicates whether conversion of a precision input 2 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
1 PIEOCF1	Precision Input End Of Conversion Flag 1 Indicates whether conversion of a precision input 1 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
0 PIEOCF0	Precision Input End Of Conversion Flag 0 Indicates whether conversion of a precision input 0 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clears flag

57.5.2.6 Channel End Of Conversion Flag For Standard Inputs (CEOCFR1)

Offset

Register	Offset
CEOCFR1	18h

Function
Contains flags that indicate whether conversion is complete for a standard input.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								SIEOCF23	SIEOCF22	SIEOCF21	SIEOCF20	SIEOCF19	SIEOCF18	SIEOCF17	SIEOCF16
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SIEOCF15	SIEOCF14	SIEOCF13	SIEOCF12	SIEOCF11	SIEOCF10	SIEOCF9	SIEOCF8	SIEOCF7	SIEOCF6	SIEOCF5	SIEOCF4	SIEOCF3	SIEOCF2	SIEOCF1	SIEOCF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 SIEOCFn	Standard Input End Of Conversion Flag Indicates whether conversion on standard input *n* is complete. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag

57.5.2.7 Channel End Of Conversion Flag For External Inputs (CEOCFR2)

Offset

Register	Offset
CEOCFR2	1Ch

Function

Contains flags that indicate whether conversion is complete for an external input.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CEOCFR2	—
ADC_1	CEOCFR2	—
ADC_2	—	CEOCFR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EIEOCF31	EIEOCF30	EIEOCF29	EIEOCF28	EIEOCF27	EIEOCF26	EIEOCF25	EIEOCF24	EIEOCF23	EIEOCF22	EIEOCF21	EIEOCF20	EIEOCF19	EIEOCF18	EIEOCF17	EIEOCF16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EIEOCF15	EIEOCF14	EIEOCF13	EIEOCF12	EIEOCF11	EIEOCF10	EIEOCF9	EIEOCF8	EIEOCF7	EIEOCF6	EIEOCF5	EIEOCF4	EIEOCF3	EIEOCF2	EIEOCF1	EIEOCF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EIEOCFn	External Input End Of Conversion Flag Indicates whether conversion on external input *n* is complete. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag

57.5.2.8 Interrupt Mask (IMR)

Offset

Register	Offset
IMR	20h

Function
Enables flagging of interrupts.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											MSKEOBC...	MSKJEOC	MSKJECH	MSKEOC	MSKECH
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4 MSKEOBCTU	EOBCTU Interrupt Flag Enable Specifies whether a completed BCTU conversion flags the EOBCTU interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
3 MSKJEOC	JEOC Interrupt Flag Enable Specifies whether completion of an injected conversion flags the JEOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
2 MSKJECH	JECH Interrupt Flag Enable Specifies whether completion of an injected conversion of a set of inputs flags the JECH interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
1 MSKEOC	EOC Interrupt Flag Enable Specifies whether completion of a normal conversion flags the EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
0 MSKECH	ECH Interrupt Flag Enable Specifies whether completion of a normal conversion of a set of inputs flags the ECH interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged

57.5.2.9 EOC Interrupt Enable For Precision Inputs (CIMR0)

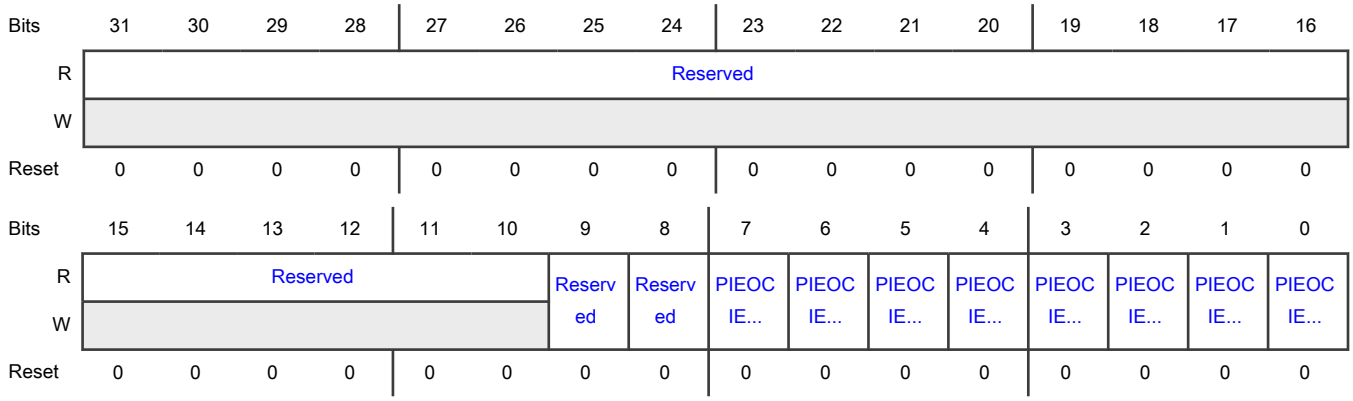
Offset

Register	Offset
CIMR0	24h

Function

Specifies whether a completed conversion of a precision input flags an EOC interrupt.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 PIEOCIEN7	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
6 PIEOCIEN6	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
5 PIEOCIEN5	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
4 PIEOCIEN4	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt is flagged
3 PIEOCIEN3	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
2 PIEOCIEN2	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
1 PIEOCIEN1	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
0 PIEOCIEN0	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged

57.5.2.10 EOC Interrupt Enable For Standard Inputs (CIMR1)

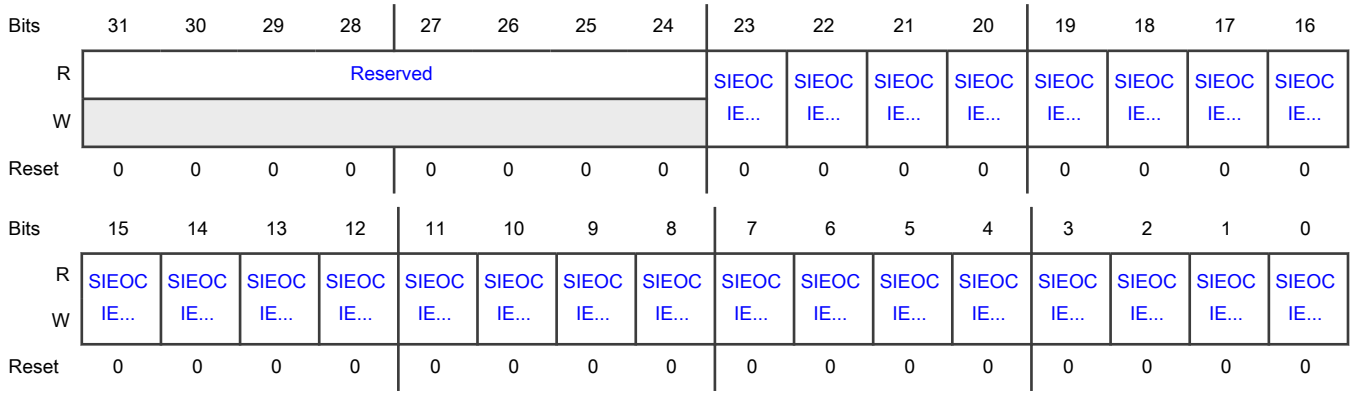
Offset

Register	Offset
CIMR1	28h

Function

Specifies whether a completed conversion of a standard input flags an EOC interrupt.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 SIEOCIENn	Standard Input EOC Interrupt Enable Specifies whether a completed conversion of standard input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged

57.5.2.11 EOC Interrupt Enable For External Inputs (CIMR2)

Offset

Register	Offset
CIMR2	2Ch

Function

Specifies whether a completed conversion of an external input flags an EOC interrupt.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CIMR2	—
ADC_1	CIMR2	—
ADC_2	—	CIMR2

Diagram

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

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

Fields

Field	Function
31-0 EIEOCIENn	External Input EOC Interrupt Enable Specifies whether a completed conversion of external input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged

57.5.2.12 Analog Watchdog Threshold Interrupt Status (WTISR)

Offset

Register	Offset
WTISR	30h

Function

Contains flags that indicate the result of a comparison between the conversion result and the threshold value defined in the analog watchdog.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HAWIF16	LAWIF16	HAWIF15	LAWIF15	HAWIF14	LAWIF14	HAWIF13	LAWIF13	HAWIF12	LAWIF12	HAWIF11	LAWIF11	HAWIF10	LAWIF10	HAWIF9	LAWIF9
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HAWIF8	LAWIF8	HAWIF7	LAWIF7	HAWIF6	LAWIF6	HAWIF5	LAWIF5	HAWIF4	LAWIF4	HAWIF3	LAWIF3	HAWIF2	LAWIF2	HAWIF1	LAWIF1
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 HAWIF16	High Analog Watchdog Interrupt Flag 16 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 16. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
30 LAWIF16	Low Analog Watchdog Interrupt Flag 16 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 16. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
29 HAWIF15	High Analog Watchdog Interrupt Flag 15 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 15. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
28	Low Analog Watchdog Interrupt Flag 15

Table continues on the next page...

Table continued from the previous page...

Field	Function
LAWIF15	Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 15. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
27 HAWIF14	High Analog Watchdog Interrupt Flag 14 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 14. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
26 LAWIF14	Low Analog Watchdog Interrupt Flag 14 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 14. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
25 HAWIF13	High Analog Watchdog Interrupt Flag 13 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 13.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
24 LAWIF13	Low Analog Watchdog Interrupt Flag 13 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 13. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
23 HAWIF12	High Analog Watchdog Interrupt Flag 12 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 12. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
22 LAWIF12	Low Analog Watchdog Interrupt Flag 12 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 12.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
21 HAWIF11	High Analog Watchdog Interrupt Flag 11 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 11. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
20 LAWIF11	Low Analog Watchdog Interrupt Flag 11 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 11. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
19 HAWIF10	High Analog Watchdog Interrupt Flag 10 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 10.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
18 LAWIF10	Low Analog Watchdog Interrupt Flag 10 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 10. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
17 HAWIF9	High Analog Watchdog Interrupt Flag 9 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 9. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
16 LAWIF9	Low Analog Watchdog Interrupt Flag 9 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 9. **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
15 HAWIF8	High Analog Watchdog Interrupt Flag 8 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 8. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
14 LAWIF8	Low Analog Watchdog Interrupt Flag 8 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 8. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
13 HAWIF7	High Analog Watchdog Interrupt Flag 7 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 7. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
12 LAWIF7	Low Analog Watchdog Interrupt Flag 7 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 7. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
11 HAWIF6	High Analog Watchdog Interrupt Flag 6 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 6. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
10 LAWIF6	Low Analog Watchdog Interrupt Flag 6 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 6. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
9 HAWIF5	High Analog Watchdog Interrupt Flag 5 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 5. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
8 LAWIF5	Low Analog Watchdog Interrupt Flag 5 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 5. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
7 HAWIF4	High Analog Watchdog Interrupt Flag 4 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 4. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clears flag
6 LAWIF4	Low Analog Watchdog Interrupt Flag 4 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 4. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
5 HAWIF3	High Analog Watchdog Interrupt Flag 3 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 3. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
4 LAWIF3	Low Analog Watchdog Interrupt Flag 3 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 3. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 HAWIF2	High Analog Watchdog Interrupt Flag 2 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 2. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
2 LAWIF2	Low Analog Watchdog Interrupt Flag 2 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 2. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
1 HAWIF1	High Analog Watchdog Interrupt Flag 1 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
0 LAWIF1	Low Analog Watchdog Interrupt Flag 1 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag

57.5.2.13 Analog Watchdog Threshold Interrupt Enable (WTIMR)

Offset

Register	Offset
WTIMR	34h

Function
Enables the interrupt for each analog watchdog threshold value.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								HDWIFEN4	LAWIFEN4	HDWIFEN3	LAWIFEN3	HDWIFEN2	LAWIFEN2	HDWIFEN1	LAWIFEN1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 HDWIFEN4	High Data Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is higher than the high threshold value of the analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
6 LAWIFEN4	Low Analog Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is lower than the low threshold value of analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
5 HDWIFEN3	High Data Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is higher than the high threshold value of the analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
4 LAWIFEN3	Low Analog Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is lower than the low threshold value of analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
3 HDWIFEN2	High Data Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is higher than the high threshold value of the analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
2 LAWIFEN2	Low Analog Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is lower than the low threshold value of analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
1 HDWIFEN1	High Data Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is higher than the high threshold value of the analog watchdog *n*.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt is not flagged 1b - Interrupt is flagged
0 LAWIFEN1	Low Analog Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is lower than the low threshold value of analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged

57.5.2.14 Direct Memory Access Configuration (DMAE)

Offset

Register	Offset
DMAE	40h

Function
Configures the DMA feature.

Diagram

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

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

Fields

Field	Function
31-2 —	Reserved
1	DMA Clear Request

Table continues on the next page...

Table continued from the previous page...

Field	Function
DCLR	Selects one of the following events to clear a DMA request: • DMA controller acknowledges the request. • Application reads the conversion data register. 0b - DMA controller acknowledges the request 1b - Conversion data register is read
0 DMAEN	DMA Enable Enables DMA. 0b - Disable 1b - Enable

57.5.2.15 DMA Request Enable For Precision Inputs (DMAR0)

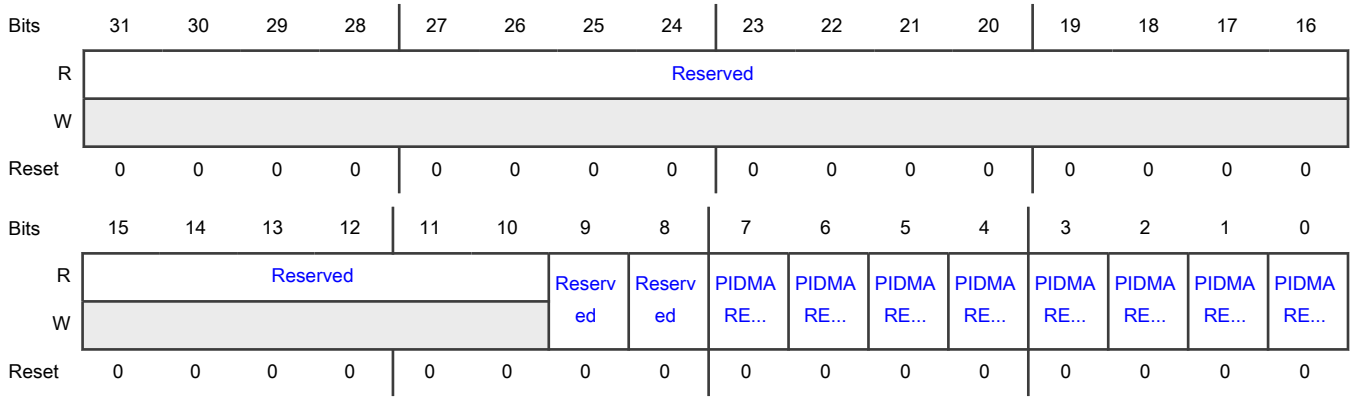
Offset

Register	Offset
DMAR0	44h

Function

Selects the precision inputs that trigger a DMA request after conversion is complete.

Diagram



Fields

Field	Function
31-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 —	Reserved
8 —	Reserved
7 PIDMAREN7	Precision Input DMA Request Enable 7 Specifies whether a DMA request is triggered after conversion is complete for precision input 7. 0b - Not triggered 1b - Triggered
6 PIDMAREN6	Precision Input DMA Request Enable 6 Specifies whether a DMA request is triggered after conversion is complete for precision input 6. 0b - Not triggered 1b - Triggered
5 PIDMAREN5	Precision Input DMA Request Enable 5 Specifies whether a DMA request is triggered after conversion is complete for precision input 5. 0b - Not triggered 1b - Triggered
4 PIDMAREN4	Precision Input DMA Request Enable 4 Specifies whether a DMA request is triggered after conversion is complete for precision input 4. 0b - Not triggered 1b - Triggered
3 PIDMAREN3	Precision Input DMA Request Enable 3 Specifies whether a DMA request is triggered after conversion is complete for precision input 3. 0b - Not triggered 1b - Triggered
2 PIDMAREN2	Precision Input DMA Request Enable 2 Specifies whether a DMA request is triggered after conversion is complete for precision input 2. 0b - Not triggered 1b - Triggered
1 PIDMAREN1	Precision Input DMA Request Enable 1 Specifies whether a DMA request is triggered after conversion is complete for precision input 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not triggered 1b - Triggered
0 PIDMAREN0	Precision Input DMA Request Enable 0 Specifies whether a DMA request is triggered after conversion is complete for precision input 0. 0b - Not triggered 1b - Triggered

57.5.2.16 DMA Request Enable For Standard Inputs (DMAR1)

Offset

Register	Offset
DMAR1	48h

Function

Selects the standard inputs that trigger a DMA request after conversion is complete.

Diagram

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

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

Fields

Field	Function
31-24 —	Reserved
23-0 SIDMARENn	Standard Input DMA Request Enable n Specifies whether a DMA request is triggered after conversion is complete for standard input *n*.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - DMA request is not triggered 1b - DMA is request triggered

57.5.2.17 DMA Request Enable For External Inputs (DMAR2)

Offset

Register	Offset
DMAR2	4Ch

Function

Selects the external inputs that trigger a DMA request after conversion is complete.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	DMAR2	—
ADC_1	DMAR2	—
ADC_2	—	DMAR2

Diagram

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

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

Fields

Field	Function
31-0	External Input DMA Request Enable n

Table continues on the next page...

Field	Function
EIDMARENn	Specifies whether a DMA request is triggered after external input *n* is converted. 0b - DMA request is not triggered 1b - DMA request is triggered

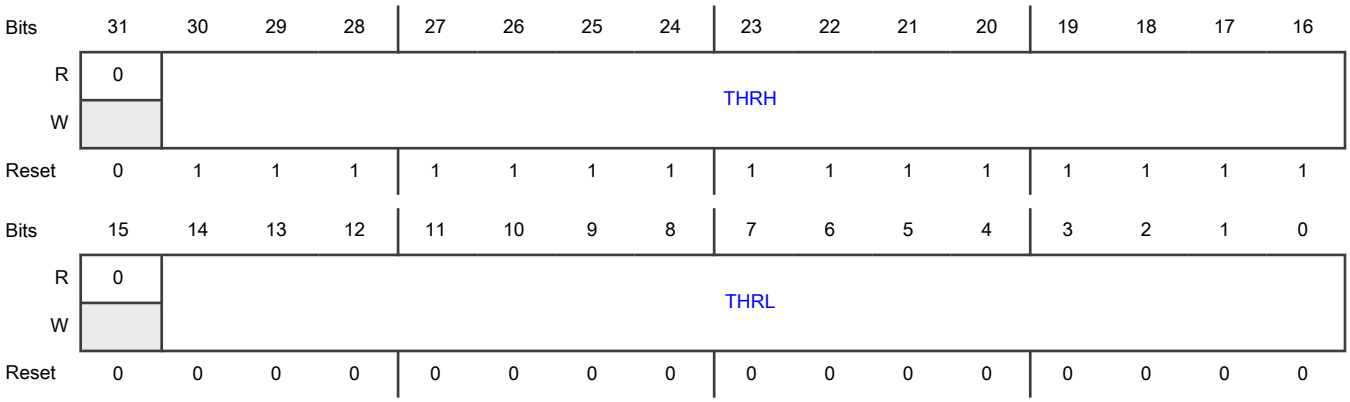
57.5.2.18 Analog Watchdog Threshold Values (THRHLR0 - THRHLR3)

Offset

Register	Offset
THRHLR0	60h
THRHLR1	64h
THRHLR2	68h
THRHLR3	6Ch

Function
Specifies limits for the valid operating range of a monitored input.

Diagram



Fields

Field	Function
31 —	Reserved
30-16 THRH	High Threshold Value If enabled, flags an interrupt when the converted data of the input is higher than this value.
15	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-0 THRL	Low Threshold Value If enabled, flags an interrupt when the converted data of the input is lower than this value.

57.5.2.19 Presampling Control (PSCR)

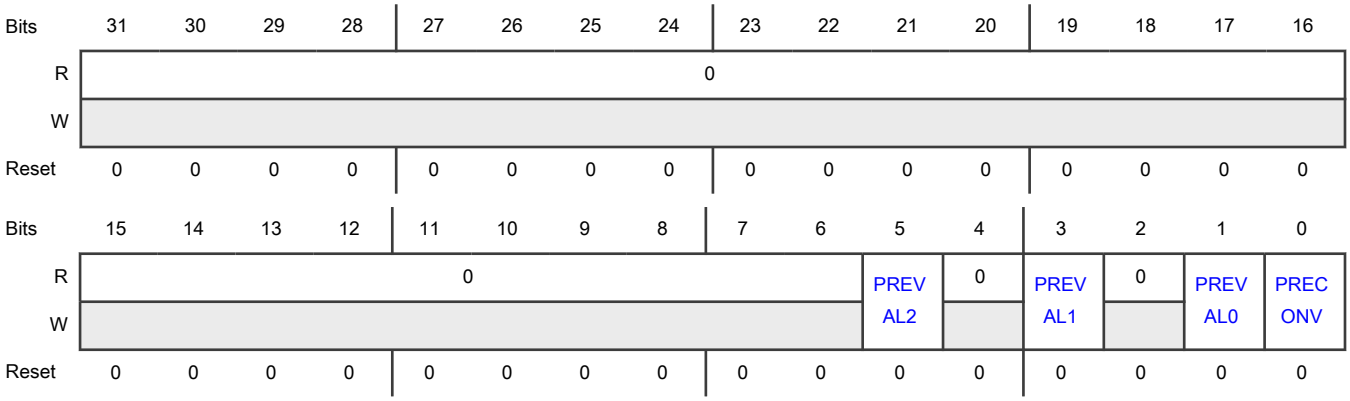
Offset

Register	Offset
PSCR	80h

Function

Configures ADC to presample an internal voltage before the actual input is converted.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 PREVAL2	Presampling Voltage Select For External Inputs Selects the internal voltage that is presampled for external inputs. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>ADC_0</td><td>PSCR</td><td>—</td></tr><tr><td>ADC_1</td><td>PSCR</td><td>—</td></tr><tr><td>ADC_2</td><td>—</td><td>PSCR</td></tr></table> 0b - VREFL 1b - VREFH
4 —	Reserved
3 PREVAL1	Presampling Voltage Select For Standard Inputs Selects the internal voltage that is presampled for standard inputs and for the temperature sensor. 0b - VREFL 1b - VREFH
2 —	Reserved
1 PREVAL0	Presampling Voltage Select For Precision Inputs Selects the internal voltage that is presampled for precision inputs. 0b - VREFL 1b - VREFH
0 PRECONV	Convert Presampled Value Specifies whether presampling is followed by the comparison. If enabled, presampling is followed by conversion and the result is written to the conversion data register of the selected input. 0b - No conversion after presampling 1b - Presampling is followed by conversion

57.5.2.20 Presampling Enable For Precision Inputs (PSR0)

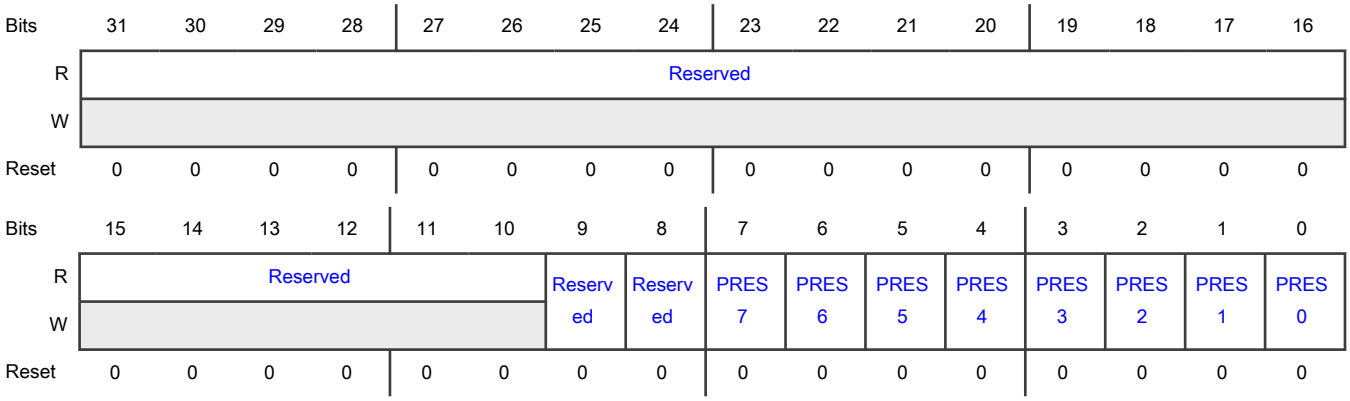
Offset

Register	Offset
PSR0	84h

Function

Enables presampling for each precision input.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 PRES7	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable
6 PRES6	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable
5 PRES5	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable
4	Presampling Enable n

Table continues on the next page...

Table continued from the previous page...

Field	Function
PRES4	Enables presampling for precision input *n*. 0b - Disable 1b - Enable
3 PRES3	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable
2 PRES2	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable
1 PRES1	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable
0 PRES0	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable

57.5.2.21 Presampling Enable For Standard Inputs (PSR1)

Offset

Register	Offset
PSR1	88h

Function

Enables presampling for each standard input.

Diagram

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

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

Fields

Field	Function
31-24 —	Reserved
23-0 PRESn	Presampling Enable n Enables presampling for standard input *n*. 0b - Disable 1b - Enable

57.5.2.22 Presampling Enable For External Inputs (PSR2)

Offset

Register	Offset
PSR2	8Ch

Function

Enables presampling for each external input.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	PSR2	—
ADC_1	PSR2	—
ADC_2	—	PSR2

Diagram

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

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

Fields

Field	Function
31-0 PRESn	Presampling Enable n Enables presampling for external input *n*. 0b - Disable 1b - Enable

57.5.2.23 Conversion Timing For Precision Inputs (CTR0)

Offset

Register	Offset
CTR0	94h

Function

Specifies duration, in terms of the number of conversion clock cycles, of the sampling of precision inputs.

The conversion clock frequency depends on the configuration of MCR[ADCLKSEL].

Diagram

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

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

Fields

Field	Function
31-8 —	Reserved
7-0 INPSAMP	Input Sample Cycles Specifies the sample duration in terms of conversion clock cycles. The minimum value is 8. Specifying a lower value automatically forces a value of 8.

57.5.2.24 Conversion Timing For Standard Inputs (CTR1)

Offset

Register	Offset
CTR1	98h

Function

Specifies the duration, in terms of the number of conversion clock cycles, of the sampling of standard inputs or the temperature sensor.

The conversion clock frequency depends on the configuration of MCR[ADCLKSEL].

Diagram

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

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

Fields

Field	Function
31-8 —	Reserved
7-0 INPSAMP	Input Sample Cycles Specifies the sample duration in terms of conversion clock cycles. The minimum value is 8. Specifying a lower value automatically forces a value of 8.

57.5.2.25 Conversion Timing For External Inputs (CTR2)

Offset

Register	Offset
CTR2	9Ch

Function

Specifies the number of conversion clock cycles when the external inputs are sampled.

The conversion clock frequency depends on the configuration of MCR[ADCLKSEL].

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CTR2	—
ADC_1	CTR2	—
ADC_2	—	CTR2

Diagram

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

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

Fields

Field	Function
31-8 —	Reserved
7-0 INPSAMP	Input Sample Cycles Specifies the sample duration in terms of conversion clock cycles. The minimum acceptable value is 8. Specifying a lower value automatically forces a value of 8.

57.5.2.26 Normal Conversion Enable For Precision Inputs (NCMR0)

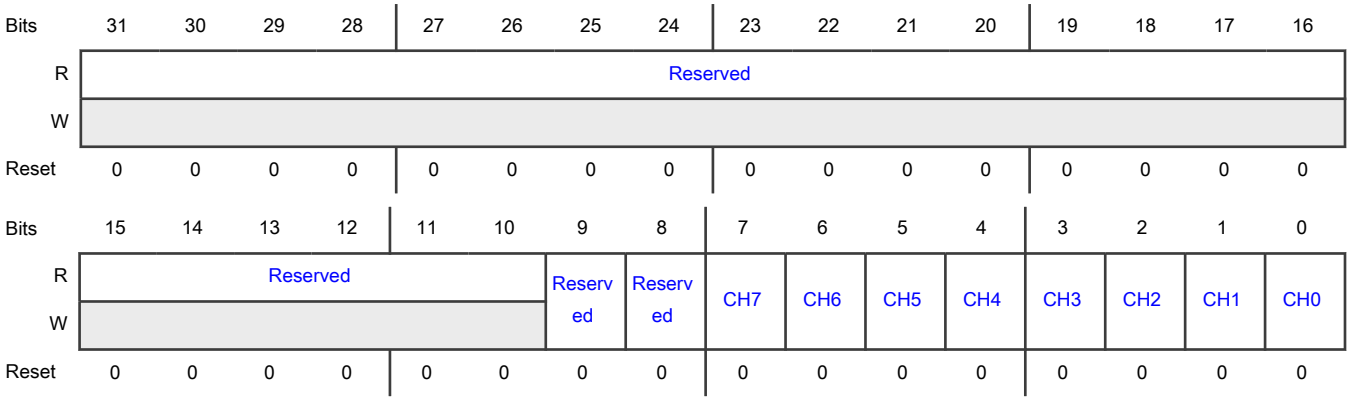
Offset

Register	Offset
NCMR0	A4h

Function

Selects the precision inputs to be converted during a normal conversion.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 CH7	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected
6 CH6	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Input is selected
5 CH5	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected
4 CH4	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected
3 CH3	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected
2 CH2	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected
1 CH1	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected
0 CH0	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected

57.5.2.27 Normal Conversion Enable For Standard Inputs (NCMR1)

Offset

Register	Offset
NCMR1	A8h

Function

Selects the standard inputs to be converted during a normal conversion.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								CH55	CH54	CH53	CH52	CH51	CH50	CH49	CH48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CH47	CH46	CH45	CH44	CH43	CH42	CH41	CH40	CH39	CH38	CH37	CH36	CH35	CH34	CH33	CH32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 CHn	Standard Input To Be Converted Selects standard input *n* for conversion. 0b - Input *n* is not selected 1b - Input *n* is selected

57.5.2.28 Normal Conversion Enable For External Inputs (NCMR2)

Offset

Register	Offset
NCMR2	ACh

Function

Selects the external inputs to be converted during a normal conversion.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	NCMR2	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
ADC_1	NCMR2	—
ADC_2	—	NCMR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CH95	CH94	CH93	CH92	CH91	CH90	CH89	CH88	CH87	CH86	CH85	CH84	CH83	CH82	CH81	CH80
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CH79	CH78	CH77	CH76	CH75	CH74	CH73	CH72	CH71	CH70	CH69	CH68	CH67	CH66	CH65	CH64
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CHn	External Input To Be Converted Selects external input *n* for conversion. 0b - Input *n* is not selected 1b - Input *n* is selected

57.5.2.29 Injected Conversion Enable For Precision Inputs (JCMR0)

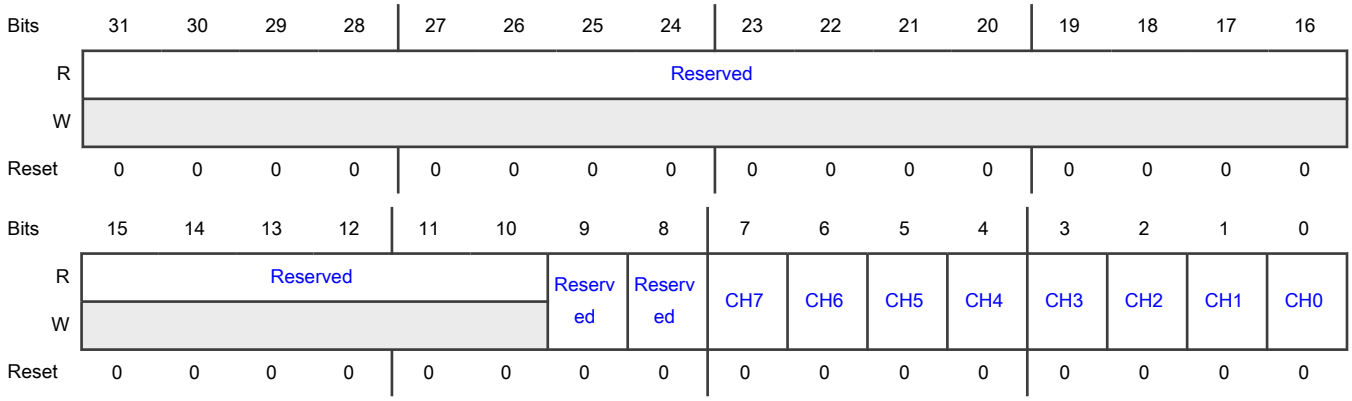
Offset

Register	Offset
JCMR0	B4h

Function

Selects the precision inputs to be converted during an injected conversion.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 CH7	Precision Input To Be Converted Selects precision input 7 for conversion. 0b - Input 7 is not selected 1b - Input 7 is selected
6 CH6	Precision Input To Be Converted Selects precision input 6 for conversion. 0b - Input 6 is not selected 1b - Input 6 is selected
5 CH5	Precision Input To Be Converted Selects precision input 5 for conversion. 0b - Input 5 is not selected 1b - Input 5 is selected
4 CH4	Precision Input To Be Converted Selects precision input 4 for conversion. 0b - Input 4 is not selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Input 4 is selected
3 CH3	Precision Input To Be Converted Selects precision input 3 for conversion. 0b - Input 3 is not selected 1b - Input 3 is selected
2 CH2	Precision Input To Be Converted Selects precision input 2 for conversion. 0b - Input 2 is not selected 1b - Input 2 is selected
1 CH1	Precision Input To Be Converted Selects precision input 1 for conversion. 0b - Input 1 is not selected 1b - Input 1 is selected
0 CH0	Precision Input To Be Converted Selects precision input 0 for conversion. 0b - Input 0 is not selected 1b - Input 0 is selected

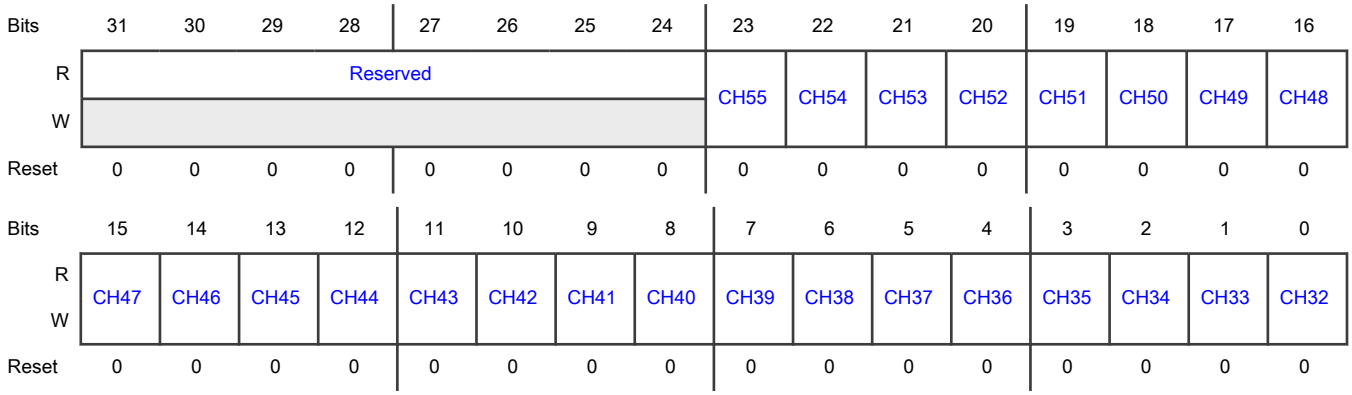
57.5.2.30 Injected Conversion Enable For Standard Inputs (JCMR1)

Offset

Register	Offset
JCMR1	B8h

Function
Selects the standard inputs to be converted during an injected conversion.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 CHn	Standard Input To Be Converted Selects standard input *n* for conversion. 0b - Input *n* is not selected 1b - Input *n* is selected

57.5.2.31 Injected Conversion Enable For External Inputs (JCMR2)

Offset

Register	Offset
JCMR2	BCh

Function

Selects the external inputs to be converted during an injected conversion.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	JCMR2	—
ADC_1	JCMR2	—
ADC_2	—	JCMR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CH95	CH94	CH93	CH92	CH91	CH90	CH89	CH88	CH87	CH86	CH85	CH84	CH83	CH82	CH81	CH80
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CH79	CH78	CH77	CH76	CH75	CH74	CH73	CH72	CH71	CH70	CH69	CH68	CH67	CH66	CH65	CH64
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CHn	External Input To Be Converted Selects external input *n* for conversion. 0b - Input *n* is not selected 1b - Input *n* is selected

57.5.2.32 Delay Start Of Data Conversion (DSDR)

Offset

Register	Offset
DSDR	C4h

Function

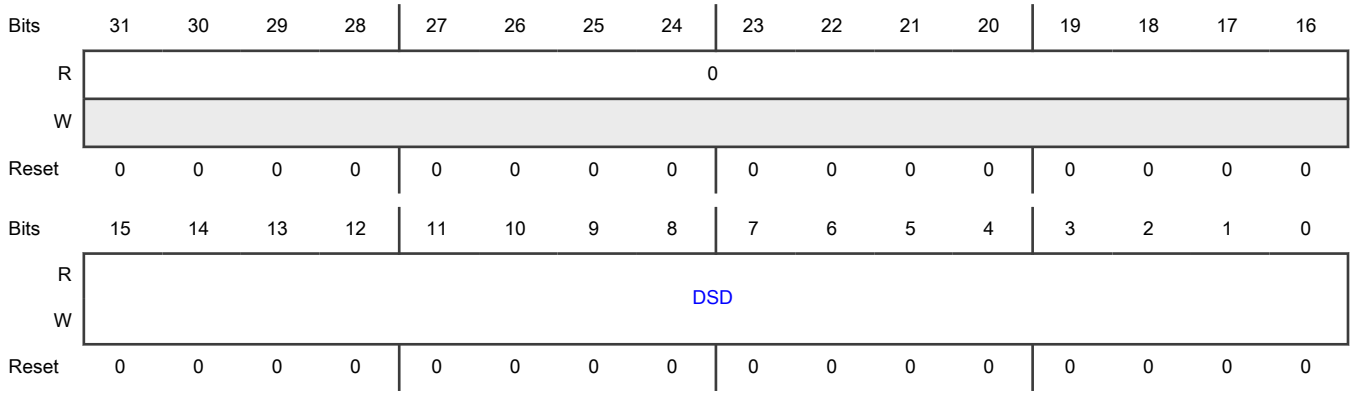
Delays the start of the conversion when another input channel is selected. The necessary delay depends on the device characteristics of, for example, the input channel multiplexer.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	DSDR	—
ADC_1	DSDR	—
ADC_2	—	DSDR

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 DSD	Delay Specifies the delay in terms of the number of module clock cycles.

57.5.2.33 Power Down Exit Delay (PDEDR)

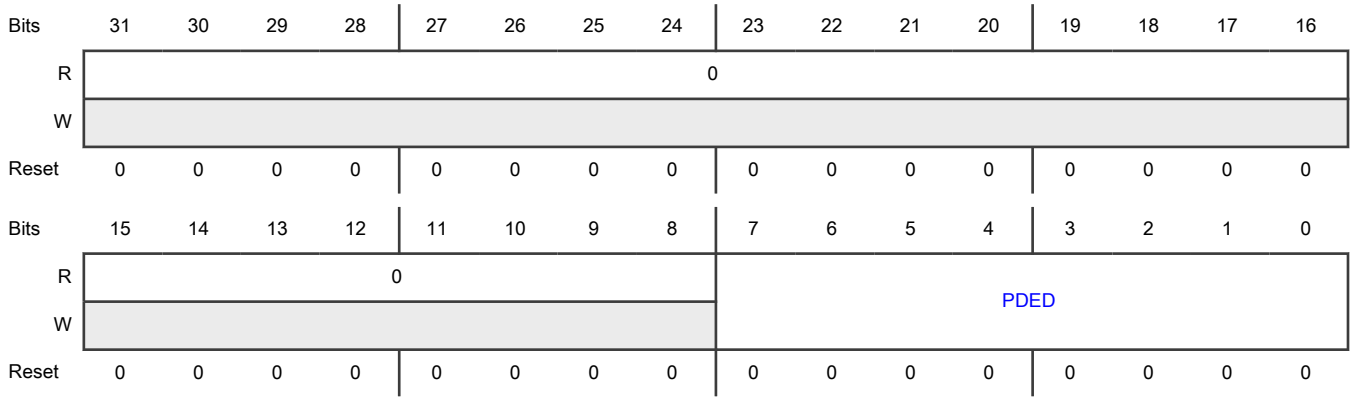
Offset

Register	Offset
PDEDR	C8h

Function

Delays the transition out of Power Down into Idle state to wait for necessary settling times. See the device data sheet for the minimal time between the power-down exit request and when ADC can start a conversion.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 PDED	Delay Specifies a delay in terms of the number of conversion clock cycles.

57.5.2.34 Precision Input n Conversion Data (PCDR0 - PCDR7)

Offset

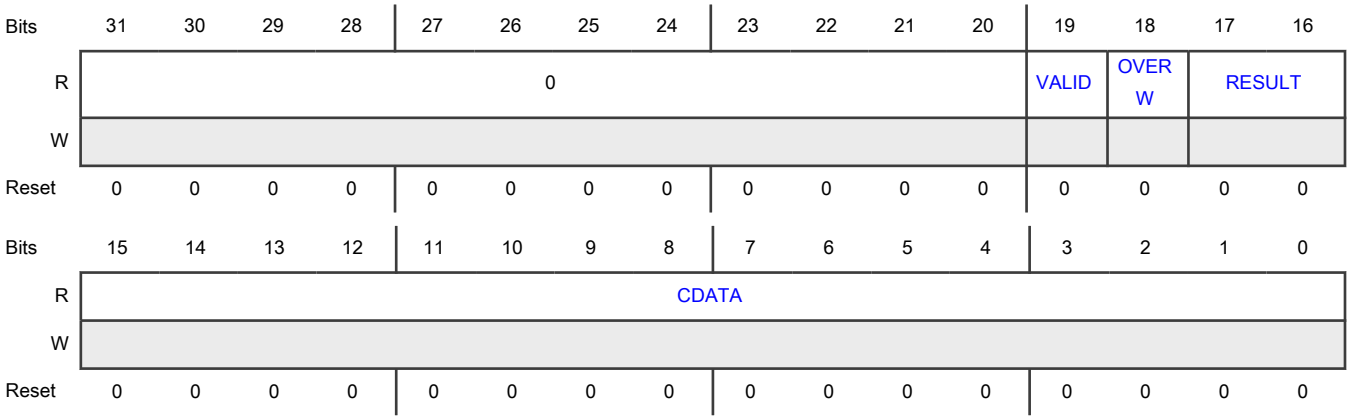
For n = 0 to 7:

Register	Offset
PCDRn	100h + (n × 4h)

Function

Contains conversion data from precision input *n*.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 VALID	Conversion Data Available Indicates whether new conversion data is available. This field is automatically reset to 0 when the data is read.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No unread conversion data 1b - Unread conversion data is available
18 OVERW	Overwrite Status Flag Indicates whether the previous conversion data was overwritten without having been read, in which case the overwritten data is lost. The ability to overwrite conversion data is controlled by MCR[OWREN]. 0b - No unread data is overwritten 1b - Unread data is overwritten
17-16 RESULT	Conversion Data Type Indicates the type of trigger that started the conversion for this conversion data. 00b - Normal trigger 01b - Injected trigger 10b - BCTU trigger
15-0 CDATA	Conversion Data Contains conversion data from precision input *n*, determined by the successive approximation algorithm. The conversion data is always 15 bits wide, regardless of the conversion resolution selected (CALBISTREG[RESN]). Depending on the value of MCR[WLSIDE], the conversion data can be in bits [14:0] (MCR[WLSIDE] = 0), or in bits [15:1] (MCR[WLSIDE] = 1).

57.5.2.35 Standard Input n Conversion Data (ICDR0 - ICDR23)

Offset

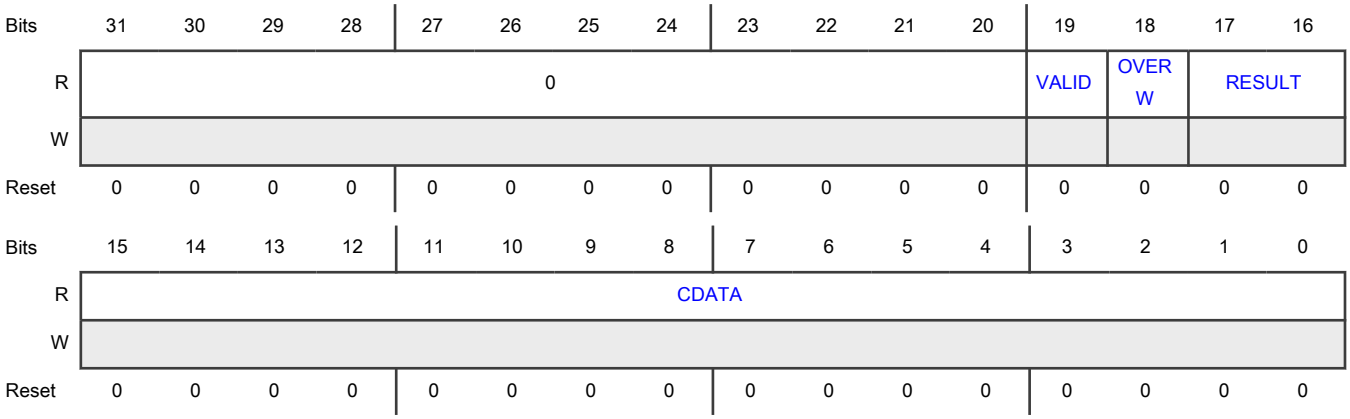
For a = 0 to 23:

Register	Offset
ICDRa	180h + (a × 4h)

Function

Contains conversion data from standard input *n*.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 VALID	Conversion Data Available Indicates whether new conversion data is available. This field is automatically reset to 0 when the data is read. 0b - No unread conversion data 1b - Unread conversion data is available
18 OVERW	Overwrite Status Flag Indicates whether the previous conversion data was overwritten without having been read, in which case the overwritten data is lost. The ability to overwrite conversion data is controlled by MCR[OWREN]. 0b - No unread data is overwritten 1b - Unread data is overwritten
17-16 RESULT	Conversion Data Type Indicates the type of trigger that started the conversion for this conversion data. 00b - Normal trigger 01b - Injected trigger 10b - BCTU trigger
15-0 CDATA	Conversion Data Contains conversion data from standard input *n*, determined by the SAR algorithm. The conversion data is always 15 bits wide, regardless of the conversion resolution selected (CALBISTREG[RESN]). Depending on the value of MCR[WLSIDE], the conversion data can be in bits [14:0] (MCR[WLSIDE] = 0), or in bits [15:1] (MCR[WLSIDE] = 1).

57.5.2.36 External Input n Conversion Data (ECDR0 - ECDR31)

Offset

For a = 0 to 31:

Register	Offset
ECDRa	200h + (a × 4h)

Function

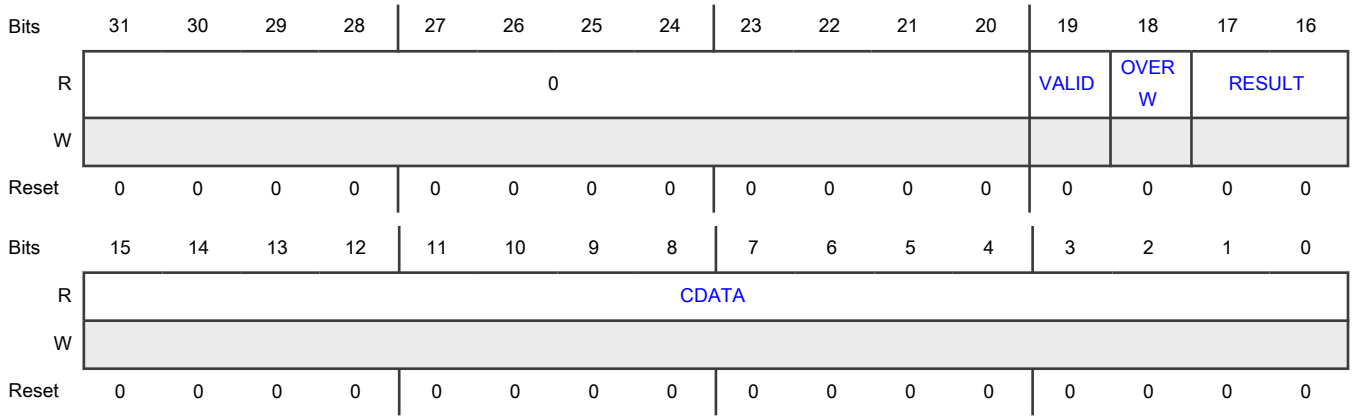
Contains conversion data from external input *n*.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	ECDR0–ECDR31	—
ADC_1	ECDR0–ECDR31	—
ADC_2	—	ECDR0–ECDR31

Diagram



Fields

Field	Function
31-20 —	Reserved
19 VALID	Conversion Data Available Indicates whether new conversion data is available. This field is automatically reset to 0 when the data is read.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No unread conversion data 1b - Unread conversion data is available
18 OVERW	Overwrite Status Flag Indicates whether the previous conversion data was overwritten without having been read, in which case the overwritten data is lost. The ability to overwrite conversion data is controlled by MCR[OWREN]. 0b - No unread data is overwritten 1b - Unread data is overwritten
17-16 RESULT	Conversion Data Type Indicates the type of trigger that started the conversion for this conversion data. 00b - Normal trigger 01b - Injected trigger 10b - BCTU trigger
15-0 CDATA	Conversion Data Contains conversion data from external input *n*, determined by the SAR algorithm. The conversion data is always 15 bits wide, regardless of the conversion resolution selected (CALBISTREG[RESN]). Depending on the value of MCR[WLSIDE], the conversion data can be in bits [14:0] (MCR[WLSIDE] = 0), or in bits [15:1] (MCR[WLSIDE] = MCR[WLSIDE]).

57.5.2.37 Channel Analog Watchdog Select For Precision Inputs (CWSELRPI0)

Offset

Register	Offset
CWSELRPI0	2B0h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor precision inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI0_7		0		WSEL_SI0_6		0		WSEL_SI0_5		0		WSEL_SI0_4	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI0_3		0		WSEL_SI0_2		0		WSEL_SI0_1		0		WSEL_SI0_0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI0_7	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI0_6	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI0_5	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI0_4	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI0_3	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI0_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI0_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI0_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

57.5.2.38 Channel Analog Watchdog Select For Precision Inputs (CWSELRPI1)

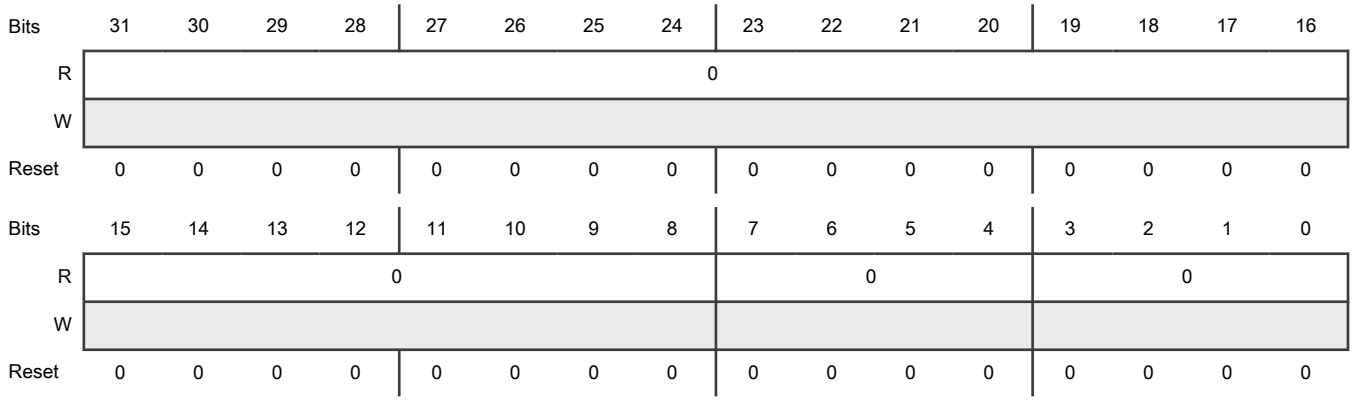
Offset

Register	Offset
CWSELRPI1	2B4h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor precision inputs.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

57.5.2.39 Channel Analog Watchdog Select For Standard Inputs (CWSELRSI0)

Offset

Register	Offset
CWSELRSI0	2C0h

Function
Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor the standard inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI7_0		0		WSEL_SI6_0		0		WSEL_SI5_0		0		WSEL_SI4_0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI3_0		0		WSEL_SI2_0		0		WSEL_SI1_0		0		WSEL_SI0_0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI7_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI6_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI5_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI4_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12	Analog Watchdog Selection

Table continues on the next page...

Table continued from the previous page...

Field	Function
WSEL_SI3_0	Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI2_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI1_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI0_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

57.5.2.40 Channel Analog Watchdog Select For Standard Inputs (CWSELRSI1)

Offset

Register	Offset
CWSELRSI1	2C4h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor the standard inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI7_1		0		WSEL_SI6_1		0		WSEL_SI5_1		0		WSEL_SI4_1	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI3_1		0		WSEL_SI2_1		0		WSEL_SI1_1		0		WSEL_SI0_1	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI7_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI6_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI5_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI4_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI3_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI2_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI1_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI0_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

57.5.2.41 Channel Analog Watchdog Select For Standard Inputs (CWSELRSI2)

Offset

Register	Offset
CWSELRSI2	2C8h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor the standard inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI7_2		0		WSEL_SI6_2		0		WSEL_SI5_2		0		WSEL_SI4_2	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI3_2		0		WSEL_SI2_2		0		WSEL_SI1_2		0		WSEL_SI0_2	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI7_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI6_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI5_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI4_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI3_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI2_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI1_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI0_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

57.5.2.42 Channel Analog Watchdog Select For External inputs (CWSELREI0)

Offset

Register	Offset
CWSELREI0	2D0h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor external inputs.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CWSELREI0	—
ADC_1	CWSELREI0	—
ADC_2	—	CWSELREI0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI0_7		0		WSEL_SI0_6		0		WSEL_SI0_5		0		WSEL_SI0_4	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI0_3		0		WSEL_SI0_2		0		WSEL_SI0_1		0		WSEL_SI0_0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI0_7	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI0_6	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI0_5	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI0_4	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI0_3	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI0_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI0_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI0_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

57.5.2.43 Channel Analog Watchdog Select For External inputs (CWSELREI1)

Offset

Register	Offset
CWSELREI1	2D4h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor external inputs.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CWSELREI1	—
ADC_1	CWSELREI1	—
ADC_2	—	CWSELREI1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI1_15		0		WSEL_SI1_14		0		WSEL_SI1_13		0		WSEL_SI1_12	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI1_11		0		WSEL_SI1_10		0		WSEL_SI1_9		0		WSEL_SI1_8	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI1_15	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI1_14	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI1_13	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI1_12	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI1_11	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI1_10	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI1_9	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI1_8	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

57.5.2.44 Channel Analog Watchdog Select For External inputs (CWSELREI2)

Offset

Register	Offset
CWSELREI2	2D8h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor external inputs.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CWSELREI2	—
ADC_1	CWSELREI2	—
ADC_2	—	CWSELREI2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI2_23		0		WSEL_SI2_22		0		WSEL_SI2_21		0		WSEL_SI2_20	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI2_19		0		WSEL_SI2_18		0		WSEL_SI2_17		0		WSEL_SI2_16	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI2_23	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI2_22	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI2_21	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI2_20	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI2_19	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI2_18	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI2_17	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI2_16	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

57.5.2.45 Channel Analog Watchdog Select For External inputs (CWSELREI3)

Offset

Register	Offset
CWSELREI3	2DCh

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor external inputs.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CWSELREI3	—
ADC_1	CWSELREI3	—
ADC_2	—	CWSELREI3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI3_31		0		WSEL_SI3_30		0		WSEL_SI3_29		0		WSEL_SI3_28	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI3_27		0		WSEL_SI3_26		0		WSEL_SI3_25		0		WSEL_SI3_24	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI3_31	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI3_30	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI3_29	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI3_28	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI3_27	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI3_26	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI3_25	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI3_24	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

57.5.2.46 Channel Watchdog Enable For Precision Inputs (CWENR0)

Offset

Register	Offset
CWENR0	2E0h

Function

Enables the analog watchdog per precision input.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						Reserved	Reserved	CWEN7	CWEN6	CWEN5	CWEN4	CWEN3	CWEN2	CWEN1	CWEN0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 CWEN7	Channel Analog Watchdog Enable 7 Enables the analog watchdog for precision input 7. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable
6 CWEN6	Channel Analog Watchdog Enable 6 Enables the analog watchdog for precision input 6. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable
5 CWEN5	Channel Analog Watchdog Enable 5 Enables the analog watchdog for precision input 5. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable
4 CWEN4	Channel Analog Watchdog Enable 4 Enables the analog watchdog for precision input 4. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable
3 CWEN3	Channel Analog Watchdog Enable 3 Enables the analog watchdog for precision input 3. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 CWEN2	Channel Analog Watchdog Enable 2 Enables the analog watchdog for precision input 2. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable
1 CWEN1	Channel Analog Watchdog Enable 1 Enables the analog watchdog for precision input 1. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable
0 CWEN0	Channel Analog Watchdog Enable 0 Enables the analog watchdog for precision input 0. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable

57.5.2.47 Channel Watchdog Enable For Standard Inputs (CWENR1)

Offset

Register	Offset
CWENR1	2E4h

Function

Enables the analog watchdog for standard input *n*.

When enabled, conversion data on the input is compared to the selected data watchdog threshold value.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								CWEN 55	CWEN 54	CWEN 53	CWEN 52	CWEN 51	CWEN 50	CWEN 49	CWEN 48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CWEN 47	CWEN 46	CWEN 45	CWEN 44	CWEN 43	CWEN 42	CWEN 41	CWEN 40	CWEN 39	CWEN 38	CWEN 37	CWEN 36	CWEN 35	CWEN 34	CWEN 33	CWEN 32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 CWENn	Channel Analog Watchdog Enable For Standard Inputs Enables the analog watchdog for standard input *n*. 0b - Disable 1b - Enable

57.5.2.48 Channel Watchdog Enable For External Inputs (CWENR2)

Offset

Register	Offset
CWENR2	2E8h

Function

Enables the analog watchdog per external input.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CWENR2	—
ADC_1	CWENR2	—
ADC_2	—	CWENR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CWEN 95	CWEN 94	CWEN 93	CWEN 92	CWEN 91	CWEN 90	CWEN 89	CWEN 88	CWEN 87	CWEN 86	CWEN 85	CWEN 84	CWEN 83	CWEN 82	CWEN 81	CWEN 80
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CWEN 79	CWEN 78	CWEN 77	CWEN 76	CWEN 75	CWEN 74	CWEN 73	CWEN 72	CWEN 71	CWEN 70	CWEN 69	CWEN 68	CWEN 67	CWEN 66	CWEN 65	CWEN 64
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CWENn	Channel Analog Watchdog Enable For External Inputs Enables the analog watchdog for external input *n*. 0b - Disable 1b - Enable

57.5.2.49 Analog Watchdog Out Of Range For Precision Inputs (AWORR0)

Offset

Register	Offset
AWORR0	2F0h

Function

Indicates the status of analog watchdog comparisons on precision inputs.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						Reserved	Reserved	AWOR_CH7	AWOR_CH6	AWOR_CH5	AWOR_CH4	AWOR_CH3	AWOR_CH2	AWOR_CH1	AWOR_CH0
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 AWOR_CH7	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 7 is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
6 AWOR_CH6	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 6 is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
5 AWOR_CH5	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 5 is out of the limits defined by the selected analog watchdog threshold value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
4 AWOR_CH4	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 4 is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
3 AWOR_CH3	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 3 is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
2 AWOR_CH2	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 2 is out of the limits defined by the selected analog watchdog threshold value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
1 AWOR_CH1	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 1 is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
0 AWOR_CH0	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 0 is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag

57.5.2.50 Analog Watchdog Out Of Range For Standard Inputs (AWORR1)

Offset

Register	Offset
AWORR1	2F4h

Function

Indicates the status of analog watchdog comparisons on standard inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH9	AWOR_CH8	AWOR_CH7	AWOR_CH6	AWOR_CH5	AWOR_CH4	AWOR_CH3	AWOR_CH2	AWOR_CH1	AWOR_CH0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 AWOR_CHn	Analog Watchdog Out Of Range For Standard Inputs Indicates whether a data conversion on a standard input *n* is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag

57.5.2.51 Analog Watchdog Out Of Range For External Inputs (AWORR2)

Offset

Register	Offset
AWORR2	2F8h

Function

Indicates the status of analog watchdog comparisons on external inputs.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	AWORR2	—
ADC_1	AWORR2	—
ADC_2	—	AWORR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH9	AWOR_CH8	AWOR_CH7	AWOR_CH6	AWOR_CH5	AWOR_CH4	AWOR_CH3	AWOR_CH2	AWOR_CH1	AWOR_CH0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 AWOR_CHn	Analog Watchdog Out Of Range For External Inputs Indicates whether a data conversion on external input *n* is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Field	Function
	0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag

57.5.2.52 Self-Test Configuration 1 (STCR1)

Offset

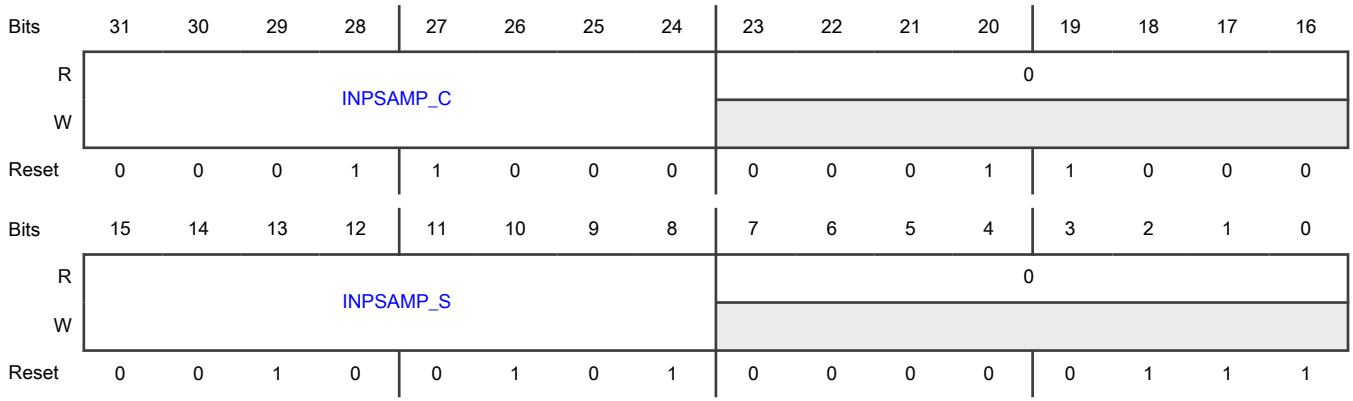
Register	Offset
STCR1	340h

Function

Specifies the input sampling duration in terms of conversion clock cycles.

Conversion clock frequency depends on the configuration of MCR[ADCLKSEL].

Diagram



Fields

Field	Function
31-24 INPSAMP_C	Input Sampling Time Algorithm C Specifies the sample duration for test conversions related to algorithm C. The minimum acceptable value is 8. Specifying a lower value automatically forces a value of 8.
23-16 —	Reserved
15-8	Input Sampling Time Algorithm S

Table continues on the next page...

Table continued from the previous page...

Field	Function
INPSAMP_S	Specifies sample duration for test conversions related to algorithm S. The minimum value is 8. Specifying a lower value automatically forces a value of 8.
7-0 —	Reserved

57.5.2.53 Self-Test Configuration 2 (STCR2)

Offset

Register	Offset
STCR2	344h

Function

Configures ADC self-test functions.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				MSKWDSE...	SERR	MSKWDTE...	0	MSKST_E...	0				MSKWDG_...	0	MSKWDG_...
W						W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MSKERR_C	0	MSKERR_...	MSKERR_...	MSKERR_...	0			EN	0		FMA_WDS...	FMA_WDT...	FMA_C	0	FMA_S
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-28 —	Reserved
27 MSKWDSERR	Mask Interrupt Self-Test Watchdog Sequence Error Generates an interrupt when the WDSERR flag transitions to 1. 0b - No interrupt is generated 1b - Interrupt is generated

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 SERR	Self-Test Error Injection Writes 1 to the STSR1[ERR_C], STSR1[ERR_S0], STSR1[ERR_S1], and STSR1[ERR_S2] status fields, causing an error. This field is reset to 0 immediately after writing. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Error can be injected 1b - Error is being injected When writing 0b - No effect 1b - Injects a self-test error
25 MSKWDTERR	Mask Interrupt Self-Test Watchdog Timer Error Generates an interrupt when the WDTERR flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
24 —	Reserved
23 MSKST_EOC	Mask Interrupt Self-Test End Of Conversion Generates an interrupt when the ST_EOC flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
22-19 —	Reserved
18 MSKWDG_EOA_C	Mask Error Interrupt End Of Algorithm C Generates an interrupt when the WDG_EOA_C flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
17 —	Reserved
16 MSKWDG_EOA_S	Mask Error Interrupt End Of Algorithm S Generates an interrupt when the WDG_EOA_S flag is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No interrupt is generated 1b - Interrupt is generated
15 MSKERR_C	Mask Error Interrupt Algorithm C Generates an interrupt when the ERR_C flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
14 —	Reserved
13 MSKERR_S2	Mask Error Interrupt Algorithm S2 Generates an interrupt when the ERR_S2 flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
12 MSKERR_S1	Mask Error Interrupt Algorithm S1 Generates an interrupt when the ERR_S1 flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
11 MSKERR_S0	Mask Error Interrupt Algorithm S0 Generates an interrupt when the ERR_S0 flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
10-8 —	Reserved
7 EN	Self-Test Enable Enables ADC structural self-test. When BCTU is enabled, this field has no effect. This field must be 1 before starting normal conversion and must not be changed during conversion. This field must be reset to 0 only after end of conversion for the last self-test channel has been received. 0b - Disable 1b - Enable
6-5 —	Reserved
4	Fault Mapping Self-Test Watchdog Sequence Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
FMA_WDSERR	Specifies whether a self-test watchdog sequence error sets the flag on the critical or noncritical fault line. 0b - Noncritical fault line 1b - Critical fault line
3 FMA_WDTERR	Fault Mapping Self-Test Watchdog Timer Error Specifies whether a self-test watchdog timer error sets the flag on the critical or noncritical fault line. 0b - Noncritical fault line 1b - Critical fault line
2 FMA_C	Fault Mapping Algorithm C Specifies whether a fault in algorithm C sets the flag on the critical or noncritical fault line. 0b - Noncritical fault line 1b - Critical fault line
1 —	Reserved
0 FMA_S	Fault Mapping Algorithm S Specifies whether a fault in algorithm S sets the flag on the critical or noncritical fault line. 0b - Noncritical fault line 1b - Critical fault line

57.5.2.54 Self-Test Configuration 3 (STCR3)

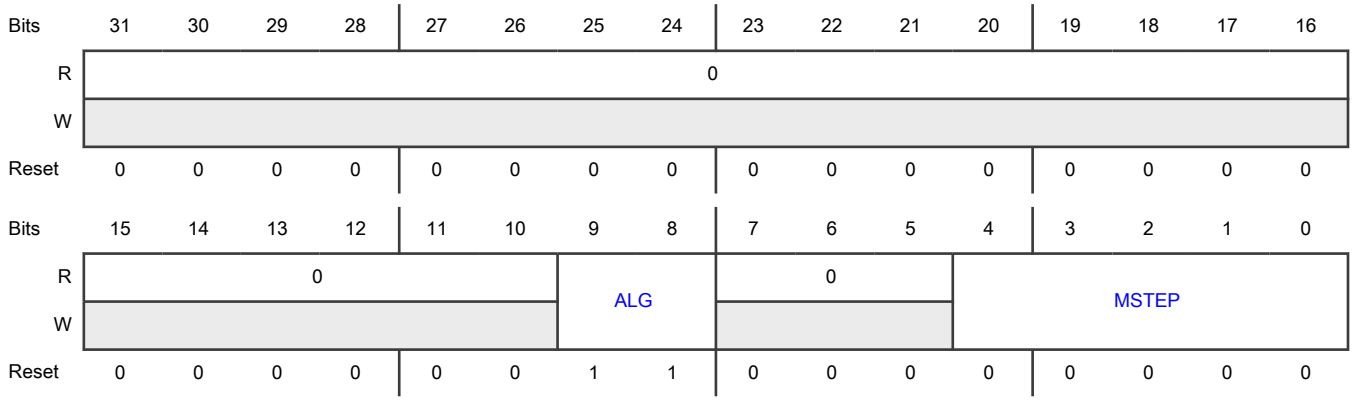
Offset

Register	Offset
STCR3	348h

Function

Configures ADC self-test functions. When BCTU is enabled, writes to the fields registers have no effect. This register must be programmed before starting a conversion and must not be changed when conversion is ongoing.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-8 ALG	Algorithm Selection One-Shot operation mode: • 00 Algorithm S (single step = MSTEP) • 01 Reserved • 10 Algorithm C (single step = MSTEP) • 11 Algorithm S (default) (use for test/debug purposes) Continuous conversion mode: • 00 Algorithm S • 01 Reserved • 10 Algorithm C • 11 Algorithm S + algorithm C (default)
7-5 —	Reserved
4-0 MSTEP	Algorithm Step For One-Shot operation mode: • MSTEP = 0 to 2 for algorithm S • MSTEP = 0 to 11 for algorithm C Unused codes are reserved and must not be used. For Scan operation mode:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is not used in Scan operation mode because single-step execution (interleaved mode) is always performed. This field must be programmed to zero in Scan operation mode.

57.5.2.55 Self-Test Baud Rate (STBRR)

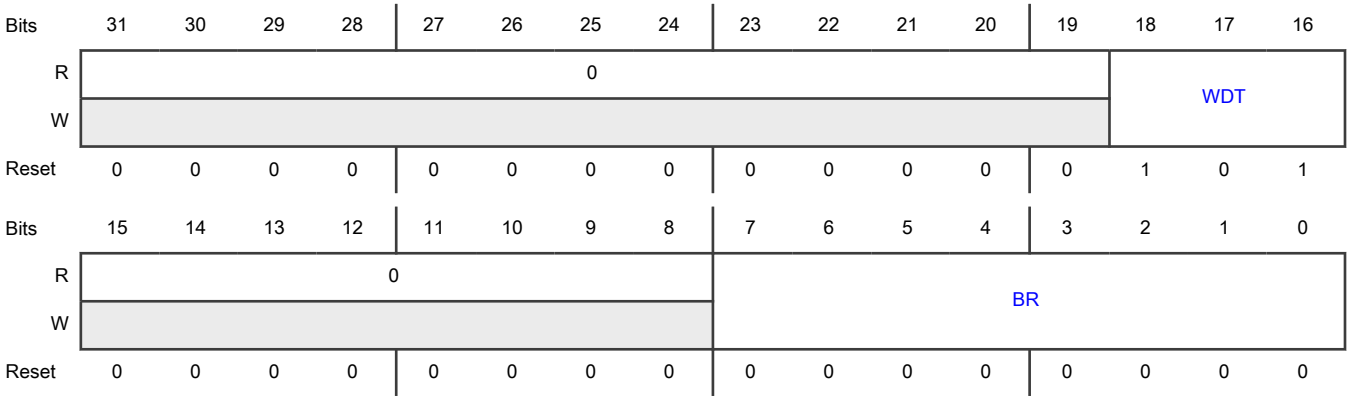
Offset

Register	Offset
STBRR	34Ch

Function

Specifies when the self-test algorithm steps are executed and the maximum time allowed for self-test to finish.

Diagram



Fields

Field	Function
31-19 —	Reserved
18-16 WDT	Self-Test Watchdog Timer Specifies the safe time period in terms of conversion clock cycles. The safe time period is the maximum duration until the running self-test is expected to finish. 000b - 8192 conversion clock cycles (~0.1 ms at 80 MHz) 001b - 39,936 conversion clock cycles (~0.5 ms at 80 MHz) 010b - 79,872 conversion clock cycles (~1 ms at 80 MHz)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - 159,744 conversion clock cycles (~2 ms at 80 MHz) 100b - 400,384 conversion clock cycles (~5 ms at 80 MHz) 101b - 799,744 conversion clock cycles (~10 ms at 80 MHz) 110b - 1,599,488 conversion clock cycles (~20 ms at 80 MHz) 111b - 3,999,744 conversion clock cycles (~50 ms at 80 MHz)
15-8 —	Reserved
7-0 BR	Baud Rate Selects the number of conversions of the selected inputs, after which the next self-test algorithm step executes. This field must be programmed when STCR2[EN] is zero, that is, before enabling self-test. 0000_0000b - A step of the selected self-test algorithm is executed every time after the set of selected inputs has been converted. 1111_1111b - A step of the selected self-test algorithm is executed after the set of selected inputs has been converted 256 times.

57.5.2.56 Self-Test Status 1 (STSR1)

Offset

Register	Offset
STSR1	350h

Function
Indicates self-test status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				WDSERR	0	WDTERR	OVERWR	ST_EOC	0				WDG_EOA...	0	WDG_EOA...
W					W1C		W1C	W1C	W1C					W1C		W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERR_C	0	ERR_S2	ERR_S1	ERR_S0	0	STEP_C						0			
W	W1C		W1C	W1C	W1C											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 WDSERR	Self-Test Watchdog Sequence Error Indicates whether the selected self-test algorithm has executed in the correct sequence. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Algorithm executed in correct sequence 1b - Algorithm did not execute in correct sequence When writing 0b - No effect 1b - Clears flag
26 —	Reserved
25 WDTERR	Self-Test Watchdog Timer Error Indicates whether the self-test algorithm has finished within the defined safe time period. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Algorithm finished within the safe time period (or safe time period not yet elapsed). 1b - Algorithm did not finish within safe time period. When writing 0b - No effect 1b - Clears flag
24 OVERWR	Self-Test Error Status Overwrite Indicates whether an error occurred when the respective error status flag (ERR_S0, ERR_S1, ERR_S2, or ERR_C) was set. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No self-test error status flag overwritten

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Self-test error status flag overwritten When writing 0b - No effect 1b - Clears flag
23 ST_EOC	Self-Test End Of Conversion Indicates whether a self-test conversion has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clears flag
22-19 —	Reserved
18 WDG_EOA_C	Self-Test Watchdog End Of Algorithm C Indicates whether self-test algorithm C has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clears flag
17 —	Reserved
16 WDG_EOA_S	Self-Test Watchdog End Of Algorithm S Indicates whether self-test algorithm S has completed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clears flag
15 ERR_C	Error Algorithm C Indicates whether an error occurred during execution of algorithm C. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clears flag
14 —	Reserved
13 ERR_S2	Error Algorithm S Step 2 Indicates whether an error occurred during execution of step 2 of algorithm S. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clears flag
12 ERR_S1	Error Algorithm S Step 1 Indicates whether an error occurred during execution of step 1 of algorithm S.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clears flag
11 ERR_S0	Error Algorithm S Step 0 Indicates whether an error occurred during execution of step 0 of algorithm S. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clears flag
10 —	Reserved
9-5 STEP_C	Step Of Algorithm C Indicates the step of self-test algorithm C in which an error occurred. Although this register field is read-only, no transfer error is generated when writing to it.
4-0 —	Reserved

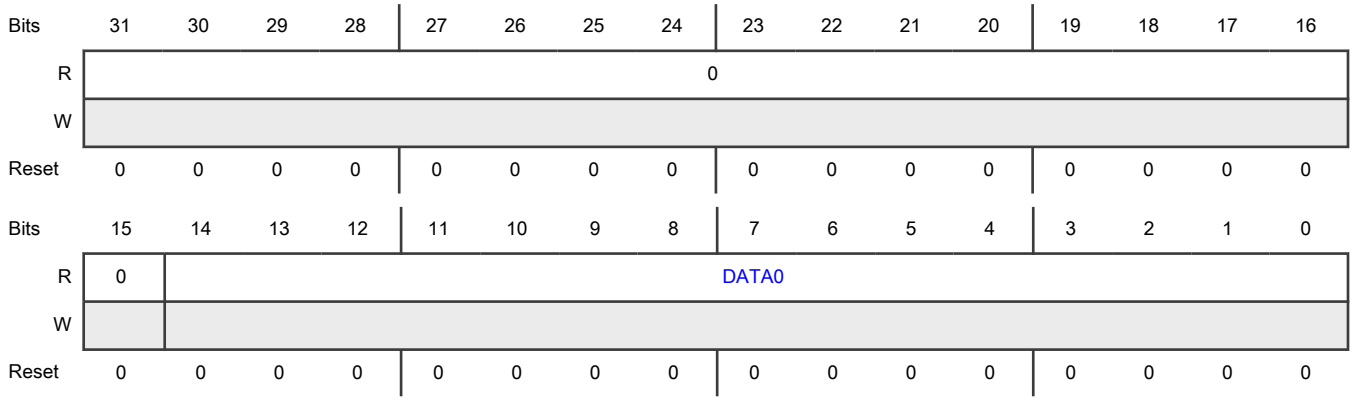
57.5.2.57 Self-Test Status 2 (STSR2)

Offset

Register	Offset
STSR2	354h

Function
Contains the conversion result when an error in step 1 of algorithm S has occurred.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 —	Reserved
14-0 DATA0	Conversion Data ERR_S1 Contains conversion data from the moment when the error in step 1 of algorithm S has occurred.

57.5.2.58 Self-Test Status 3 (STSR3)

Offset

Register	Offset
STSR3	358h

Function

Contains the conversion result when an error has occurred in step 2 or step 0 of algorithm S.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	DATA1														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	DATA0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-16 DATA1	Conversion Data ERR_S2 Contains conversion data from the moment when the error in step 2 of algorithm S has occurred.
15 —	Reserved
14-0 DATA0	Conversion Data ERR_S0 Contains the conversion data from the moment when the error in step 0 of algorithm S has occurred.

57.5.2.59 Self-Test Status 4 (STSR4)

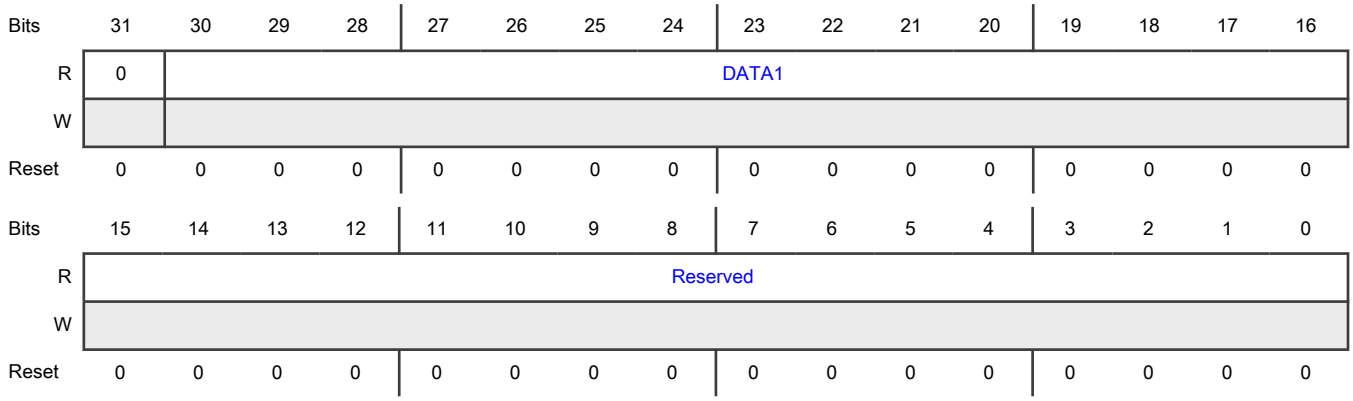
Offset

Register	Offset
STSR4	35Ch

Function

Contains the conversion result when an error has occurred in algorithm C.

Diagram



Fields

Field	Function
31 —	Reserved
30-16 DATA1	Conversion Data ERR_C Conversion data from the moment when the error in algorithm C has occurred.
15-0 —	Reserved

57.5.2.60 Self-Test Conversion Data 1 (STDR1)

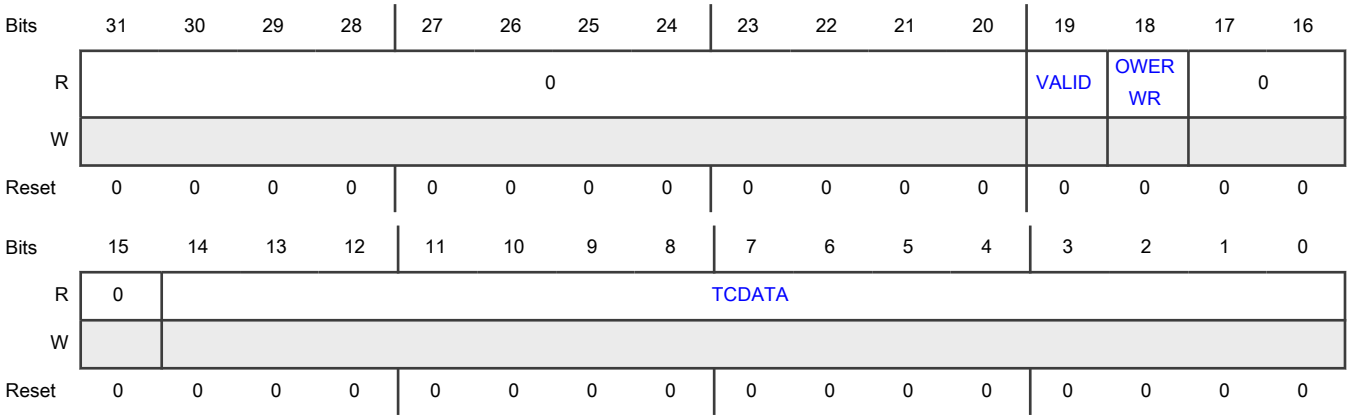
Offset

Register	Offset
STDR1	370h

Function

Contains the result of the self-test conversion.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 VALID	Valid Conversion Data Indicates that conversion data is available in TCDATA. This field is automatically reset to 0 when the conversion data is read. 0b - No unread conversion data 1b - Unread conversion data is available
18 OWERWR	Conversion Data Overwrite Status Indicates whether the conversion data in TCDATA has been overwritten over previous data that had not been read. 0b - Current conversion data not overwritten 1b - Current conversion data was overwritten
17-16 —	Reserved
15 —	Reserved
14-0 TCDATA	Test Channel Conversion Data Contains the conversion data from self-test. If overwrite is enabled (MCR[OWREN]=1), the conversion data is from the last-executed self-test conversion. When overwrite is not enabled, conversion data is available (not overwritten) until it is read by software. The conversion data in this register is in two's complement format.

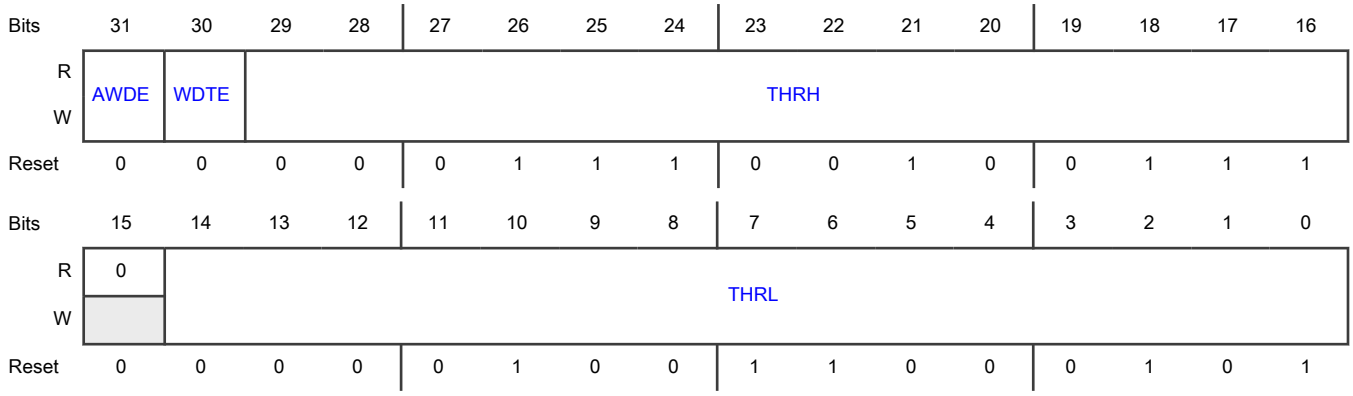
57.5.2.61 Self-Test Analog Watchdog S0 (STAW0R)

Offset

Register	Offset
STAW0R	380h

Function
Configures self-test for step 0 of algorithm S.

Diagram



Fields

Field	Function
31 AWDE	Self-Test Watchdog Enable Enables the self-test watchdog for step 0 of algorithm S. 0b - Disable 1b - Enable
30 WDTE	Self-Test Watchdog Timer Enable Enables the self-test watchdog timer for algorithm S. The self-test watchdog timer verifies: • Correct sequence of execution of the steps of the algorithm • Execution of the algorithm within the safe time period as defined by STBRR[WDT] This field must be 1 only in continuous conversion mode (MCR[MODE]=1). The safe time period starts when this field is written with 1. If the safe time period has elapsed before algorithm S starts, the WDTERR flag is set. The safe time period is restarted each time algorithm S starts. 0b - Disable 1b - Enable
29-16	Higher Threshold Value

Table continues on the next page...

Table continued from the previous page...

Field	Function
THRH	Sets the ERR_S0 flag when a self-test conversion value is greater than this value.
15 —	Reserved
14-0 THRL	Lower Threshold Value Sets the ERR_S0 flag when a self-test conversion value is less than this value.

57.5.2.62 Self-Test Analog Watchdog S1 (STAW1R)

Offset

Register	Offset
STAW1R	388h

Function

Configures self-test for step 1 of algorithm S.

Diagram

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

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

Fields

Field	Function
31 AWDE	Self-Test Watchdog Enable Enables the self-test watchdog for step 1 of algorithm S. 0b - Disable 1b - Enable
30-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 —	Reserved
14-0 THRL	Lower Threshold Value Sets the ERR_S1 flag when a self-test conversion value is less than this value.

57.5.2.63 Self-Test Analog Watchdog S2 (STAW2R)

Offset

Register	Offset
STAW2R	38Ch

Function

Configures self-test for step 2 of algorithm S.

Diagram

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

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

Fields

Field	Function
31 AWDE	Self-Test Watchdog Enable Enables the self-test watchdog for step 2 of algorithm S. 0b - Disable 1b - Enable
30-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 —	Reserved
14-0 THRL	Lower Threshold Value Sets the ERR_S2 flag when a self-test conversion value is less than this value.

57.5.2.64 Self-Test Analog Watchdog C0 (STAW4R)

Offset

Register	Offset
STAW4R	394h

Function

Configures self-test for step 0 of algorithm C.

Diagram

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

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

Fields

Field	Function
31 AWDE	Self-Test Watchdog Enable Enables the self-test watchdog for algorithm C. 0b - Disable 1b - Enable
30	Self-Test Watchdog Timer Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
WDTE	Enables the self-test watchdog timer for algorithm C. The self-test watchdog timer verifies: • Correct sequence of execution of algorithm steps • Execution of the algorithm within the safe time period as defined by STBRR[WDT] This field must be 1 only in continuous conversion mode (MODE 1). The safe time period starts when this field is written with 1. If the safe time period elapses before algorithm C starts, the WDTERR flag is set. The safe time period is restarted each time algorithm C starts. 0b - Disable 1b - Enable
29-16 THRH	Higher Threshold Value Sets the ERR_C flag when a self-test conversion value is greater than this value. This value must be in two's complement format and must be positive. This value is valid only for step 0 of algorithm C (see STAW5R for the values used for the other steps of algorithm C).
15 —	Reserved
14-0 THRL	Lower Threshold Value Sets the ERR_C flag when a self-test conversion value is less than this value. This value must be in two's complement format and must be negative. (With the reset value of this bit field, the test always fails. You must set at least STAW4R[14]=1, in order to have a negative value in this bit field.) This value is used only for step 0 of algorithm C (see STAW5R for the values used for the other steps of algorithm C).

57.5.2.65 Self-Test Analog Watchdog C (STAW5R)

Offset

Register	Offset
STAW5R	398h

Function

Configures self-test for steps 1–11 of algorithm C.

Diagram

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

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

Fields

Field	Function
31 —	Reserved
30-16 THRH	Higher Threshold Value Sets the ERR_C flag when a self-test conversion value is greater than this value. This value must be entered in two's complement format and must be positive. This value is valid only for steps 1–11 of algorithm C (see STAW4R for the value used for step 0 of algorithm C).
15 —	Reserved
14-0 THRL	Lower Threshold Value Sets the ERR_C flag when a self-test conversion value is less than this value. This value must be in two's complement format and must be negative. This value is used only for steps 1–11 of algorithm C (see STAW4R for the value used for step 0 of algorithm C).

57.5.2.66 Analog Miscellaneous In/Out register (AMSIO)

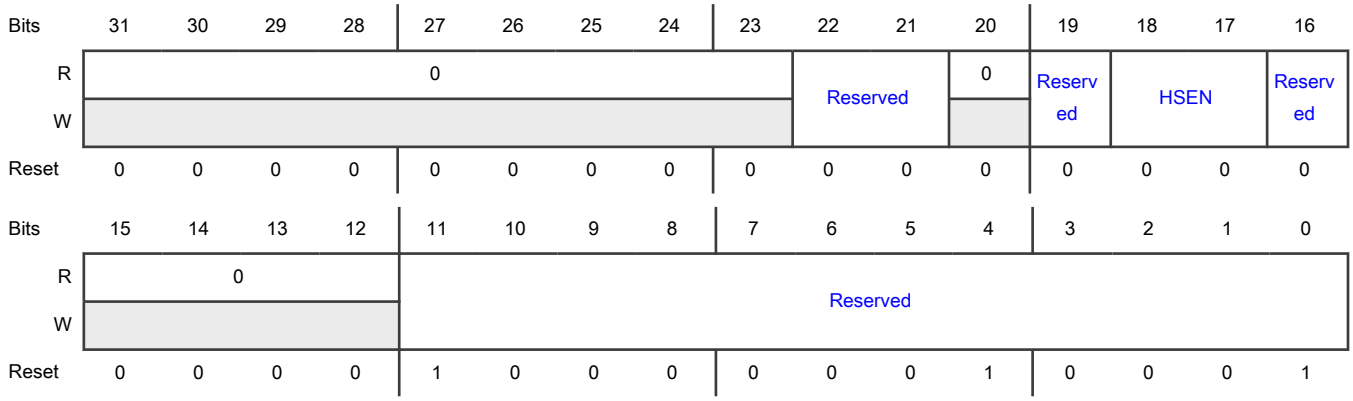
Offset

Register	Offset
AMSIO	39Ch

Function

Configures the SAR algorithm compare step. In case you want to do a high-speed calibration or a high-speed conversion (see Table 273), you must write 3d to HSEN. All other values in this register must stay at their reset value and may not be written.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-21 —	Reserved
20 —	Reserved
19 —	Reserved
18-17 HSEN	High-Speed Enable Enables high-speed conversion or calibration when set. All other bits must always keep their reset value. 00b - Normal conversion speed 11b - High-speed conversion All other values - reserved
16 —	Reserved
15-12 —	Reserved
11-0 —	Reserved

57.5.2.67 Control And Calibration Status (CALBISTREG)

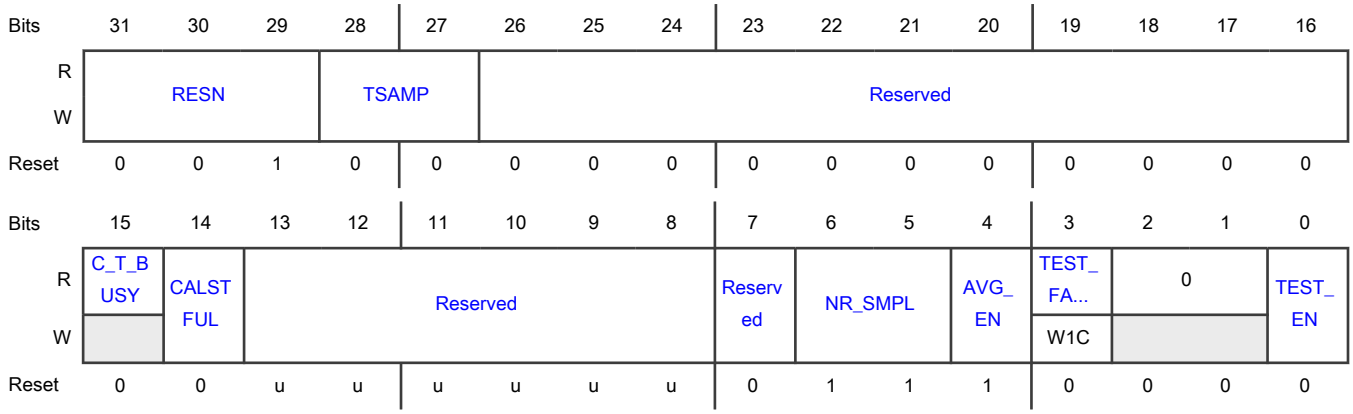
Offset

Register	Offset
CALBISTREG	3A0h

Function

Controls several ADC functions for data conversion, calibration, and calibration status.

Diagram



Fields

Field	Function
31-29 RESN	Conversion Resolution Specifies the number of significant bits per conversion data (functional conversion only, not for calibration or self-test). Reducing this number speeds up the conversion because the SAR algorithm executes fewer steps. This field must be modified only when ADC is idle. 000b - 14-bit resolution 001b - 12-bit resolution 010b - 10-bit resolution 011b - 8-bit resolution
28-27 TSAMP	Sample Period In Calibration Specifies the number of conversion clock cycles during which the reference voltage is sampled. 00b - 22 conversion clock cycles 01b - 8 conversion clock cycles 10b - 16 conversion clock cycles 11b - 32 conversion clock cycles

Table continues on the next page...

Table continued from the previous page...

Field	Function
26-16 —	Reserved The value of this field must not be modified.
15 C_T_BUSY	Calibration Busy Indicates whether calibration is running. 0b - Calibration can be started 1b - Calibration is in progress
14 CALSTFUL	Calibration And Self-Test Full Range Comparison Specifies the range of conversion data bits compared when ADC is calibrated or running self-test. Because the results from calibration and self-test are so low that the significant bits are in the lower 11 bits only, the comparison cycles for the upper 4 bits can be skipped to reduce the execution duration of the calibration and of the self-test. 0b - Lowest 11 bits are compared. 1b - All 15 bits are compared.
13-8 —	Reserved **CAUTION** This field contains a value that ADC uses internally. Writing to this field can cause ADC to operate unreliably.
7 —	Reserved This field is reserved and always has the value 0. Only the reset value can be written to this field.
6-5 NR_SMPL	Calibration Averaging Number Specifies the number of data conversions per result (averaging) during calibration. 00b - 4 samples 01b - 8 samples 10b - 16 samples 11b - 32 samples
4 AVG_EN	Calibration Averaging Enable Enables averaging of the calibration results over several conversion iterations to improve accuracy. This field only affects the calibration process. 0b - Disable 1b - Enable
3 TEST_FAIL	Calibration Status Indicates calibration status. This field reads 0 when calibration is in progress. This field is reset to 0 when calibration starts.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Calibration finished successfully or has not been run since the last reset 1b - Calibration did not finish successfully When writing 0b - No effect 1b - Clears flag
2-1 —	Reserved
0 TEST_EN	Calibration Enable Calibrates ADC. This field automatically resets to 0 after the calibration is complete or is interrupted. 0b - Wait to start a calibration 1b - Start calibration

57.5.2.68 Offset And Gain User (OFSGNUSR)

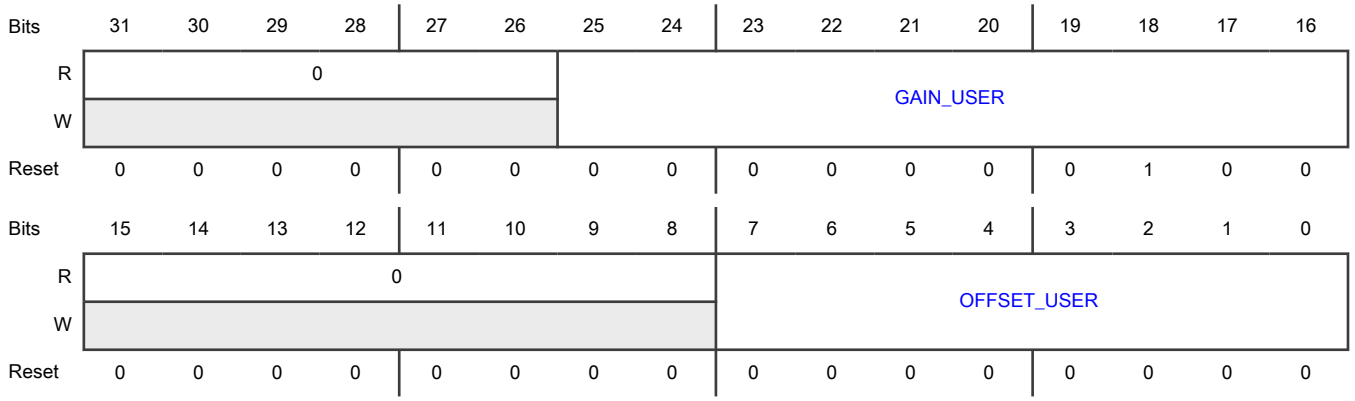
Offset

Register	Offset
OFSGNUSR	3A8h

Function

Specifies the user-configured offset and gain values used by the SAR algorithm. The values must be written in two's complement format and can be either positive or negative.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 GAIN_USER	Gain User Adds extra gain to the gain calculated during calibration. The value must be in two's complement format.
15-8 —	Reserved
7-0 OFFSET_USER	Offset User Adds extra offset to the offset calculated by calibration. The value is must be in two's complement format.

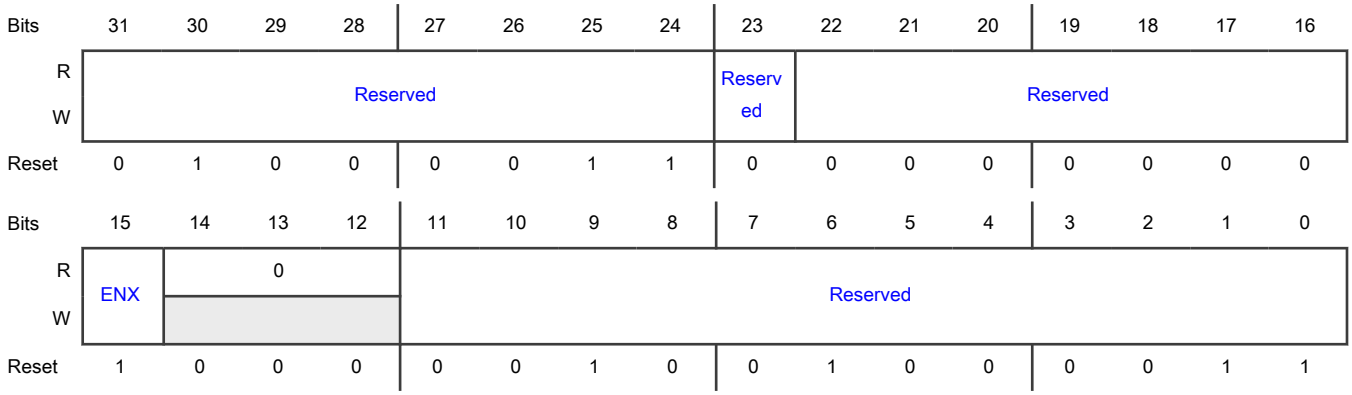
57.5.2.69 Calibration Value 2 (CAL2)

Offset

Register	Offset
CAL2	3B4h

Function
Contains fields used during calibration.

Diagram



Fields

Field	Function
31-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 —	Reserved
22-16 —	Reserved
15 ENX	Enable X Enables the inclusion of CLPX in error processing. 0b - Disable 1b - Enable
14-12 —	Reserved
11-0 —	Reserved

57.6 Glossary

CDAC	Capacitive digital to analog converter
CT	Compare phase time
DMA	Direct memory access
DP	Data processing time
PST	Presample phase time
ST	Sample phase time
SAR	Successive approximation
TPT	Trigger processing time
VrefH	Reference voltage high
VrefL	Reference voltage low

Chapter 58
Low Power Comparator (LPCMP)

58.1 Chip-specific LPCMP information

58.1.1 Instantiation information

Table 279. LPCMP instances

Chip	Instance	No. of external inputs
MWCT2D17S	LPCMP_0	8
	LPCMP_1	4
	LPCMP_2	4
MWCT2016S, MWCT2D16S	LPCMP_0	8
	LPCMP_1	4
MWCT2015S	LPCMP_0	8

58.1.2 LPCMP input output connections

See the IOMUX file attached to this document for pin/pad assignments corresponding to CMP pins.

The LPCMP channels can be used as a wakeup source in trigger mode to wakeup the device from standby mode.

58.1.3 LPCMP-DAC "vrefh0" and "vrefh1" references

The 8-bit DAC sub-block supports selection of "vrefh0" and "vrefh1" by CMPx_DCR[VRSEL]. For this device, the references are connected as follows:

- vrefh0 (External Reference): VDD_HV_A
- vrefh1 (Internal Reference): 1.2 V PMC bandgap reference

NOTE

1.2 V internal reference voltage is not available in Standby mode.

58.1.4 LPCMP window control

The window mode operation of all the comparator instances in the chip can be enabled/disabled by the TRGMUX. See the TRGMUX connectivity file attached to this document for details.

NOTE

Window signal must be of minimum 4 cycle pulse for window mode to function.

58.1.5 Comparator Trigger Mode

The comparator modules in the device support trigger mode operation as described in 'Trigger Mode' section. Device can operate in trigger mode in both standby and run mode to continuously scan the input channels. The main features of the device trigger mode operation are:

- Round robin clock: RTC_CLK
- Round robin trigger source: RTC_API

NOTE

It must be ensured that the RTC_CLK period is greater than the comparison time corresponding to the value of C0[PMODE]. It is also required to **not** select the internal reserved channels, if available on the package, for trigger mode operation by INPSEL and INNSEL. See the IOMUX file attached to this document for the pins available in various packages supported for the device.

NOTE

Only SIRC and SXOSC are supported as RTC_CLK for trigger mode operation. The LPCMP initialization delay is not supported by RRCR0[RR_INITMOD] with FXOSC or FIRC as RTC_CLK.

NOTE

In run mode, the generated trigger is delayed by 3 RTC_CLK cycles before being given to comparator trigger input.

58.1.6 Interaction with RTC API to cause wakeup

LPCMP can be used for waking up the chip from standby. For this, RTC-API and LPCMP must be configured before entering into standby mode as per below shown figure.

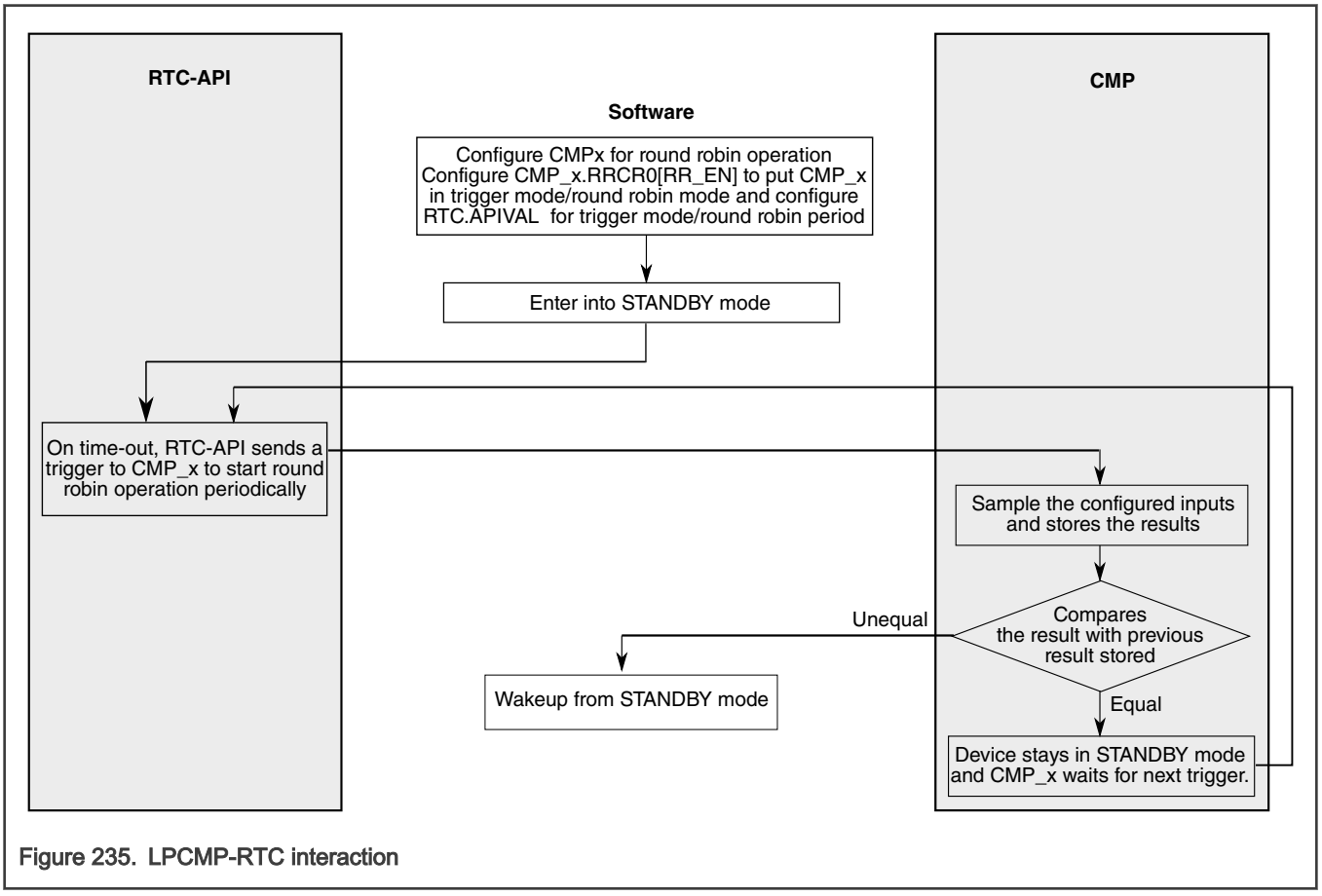


Figure 235. LPCMP-RTC interaction

In the trigger mode operation, only the continuous mode is supported. None of the window/filter/sample functions should be used. Refer to the "Functional Modes" section for details on comparator modes of operation.

Register configurations before entering Standby mode for LPCMP trigger mode operation:

1. Configure RTC.APIVAL to set the period of the round robin operation.
2. Execute standby mode entry.

58.2 Overview

The Low Power Comparator (LPCMP) module provides a circuit for comparing two analog input voltages. It comprises a comparator (CMP), a DAC and an analog mux (ANMUX). See Block diagram for details.

CMP can operate across the full range of the supply voltage, known as rail-to-rail operation.

DAC is a 256-tap resistor ladder network that provides a selectable voltage reference for applications requiring a voltage reference. The 256-tap resistor ladder network divides the supply reference V_{in} into 256 voltage levels. A 8-bit digital signal input selects the output voltage level, which varies from V_{in} to $V_{in}/256$. V_{in} can be selected from two voltage sources, vrefh0 and vrefh1. See the chip-specific LPCMP information for the source of vrefh0 and vrefh1.

NOTE

The LPCMP's internal DAC output is available as an on-chip internal signal only and is not available for an external device pin.

ANMUX allows software to select an analog input signal from among eight channel options. One channel option is the DAC output. Other device resources are connected to the other channels; see the chip-specific LPCMP information section for details. ANMUX can operate across the full range of the supply voltage.

58.2.1 Block diagram

The following figure shows the LPCMP block diagram includes the CMP, DAC, and ANMUX modules.

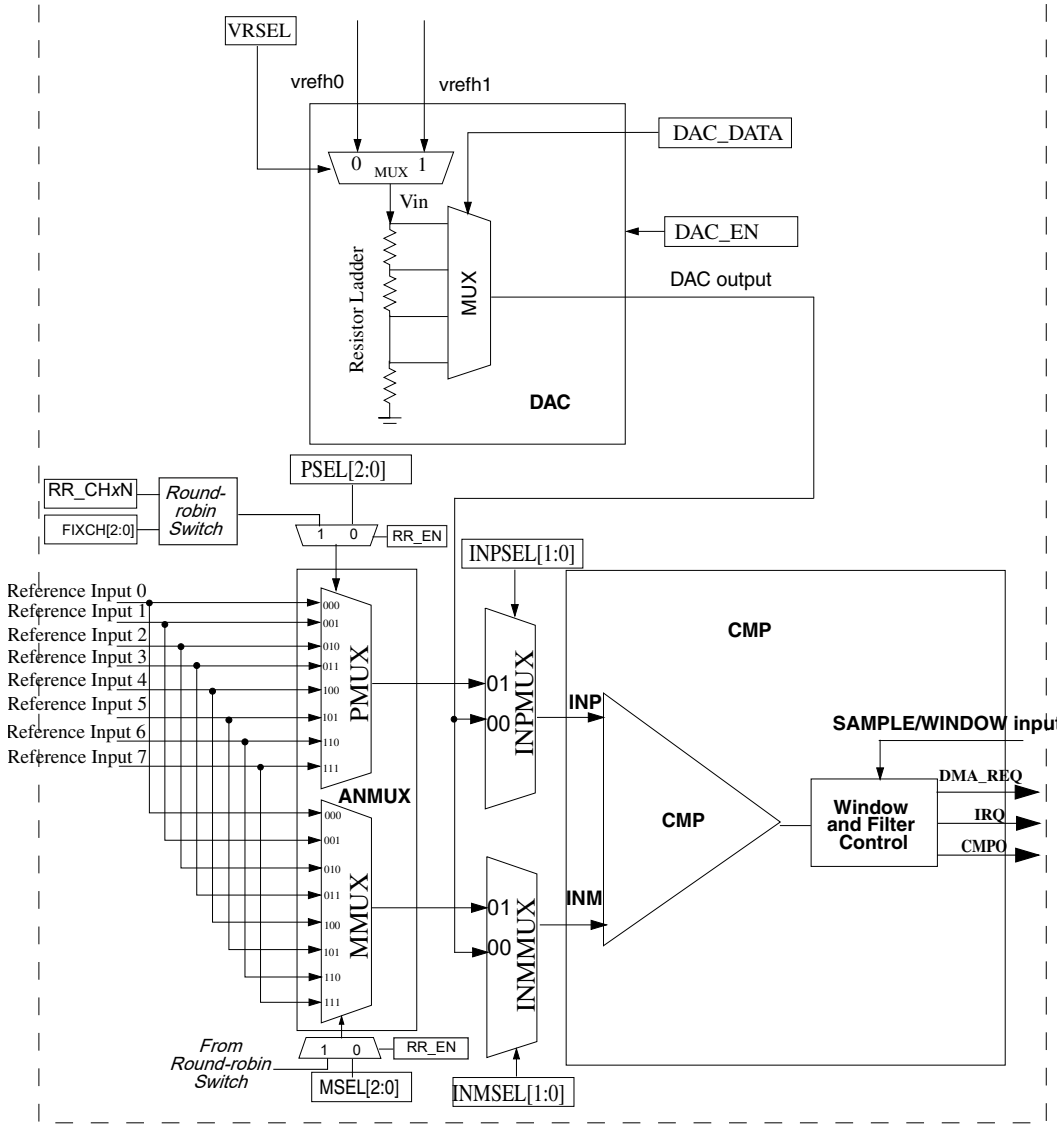


Figure 236. Block Diagram

58.2.2 Features

The features of the LPCMP module include:

- Two 8-to-1 channel MUXes to select input signal from eight channels
- Multiple operation modes to produce a wide range of outputs such as:
  - Sampled
  - Windowed, which is ideal for certain PWM zero-crossing-detection applications
  - Digitally Filtered
- Advanced feature for window and sample
  - WINDOW/SAMPLE signal can be inverted
  - Window can be closed by CMPO rising, falling or both edges

— User can define CMPO level, when window is closed

- Selectable performance levels: low power(speed) mode, high power(speed) mode
- Programmable hysteresis control
- Selectable inversion on comparator output
- External hysteresis can be used at the same time that the output filter is used for internal functions
- Interrupt and DMA support
- Trigger mode
- Includes a 8-bit resolution DAC
- Selectable supply reference source for DAC

58.3 Functional Description

58.3.1 Functional Block Diagram

The following figure shows the functional block diagram.

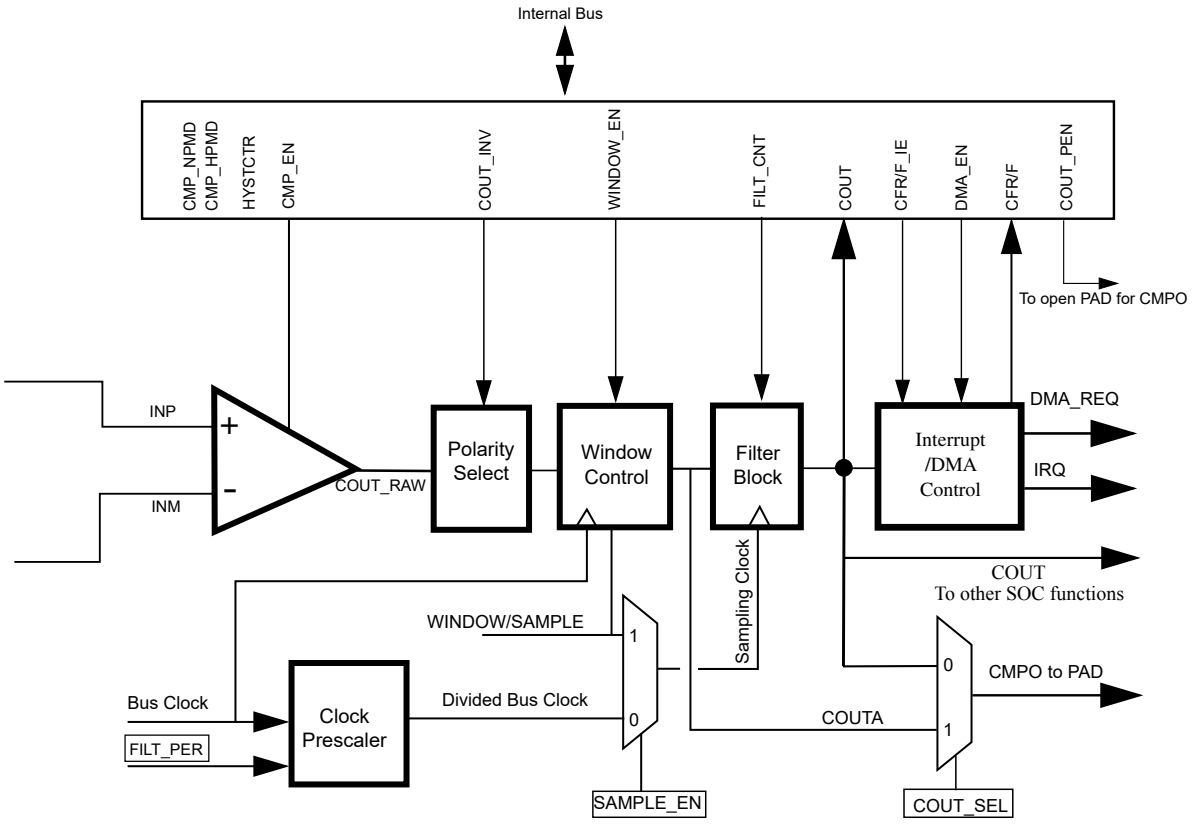


Figure 237. Functional Block Diagram

In the block diagram:

- Two analog input voltages applied to INP and INM are compared. COUT_RAW is high when the INP input voltage is greater than the INM input, and is low when the INP input voltage is less than the INM input.
- This COUT_RAW signal can be inverted if CCR1[COUT_INV] is enabled.

- If CCR1[WINDOW_EN] is enabled, the optionally inverted comparator output COUT_RAW is sampled on every bus clock when window is enabled to generate COUTA. Sampling does NOT occur when window is disabled. This feature can be used to ignore the comparator output during time periods in which the input voltages are not valid. This is especially useful when implementing zero-crossing-detection for certain PWM applications.
- The window control block can be bypassed if CCR1[WINDOW_EN] is disabled.
- The filter block acts as a simple sampler if CCR1[FILT_CNT] is set to 0x01.
- The filter block acts as a filter based on multiple samples when CCR1[FILT_CNT] is set greater than 0x01.
  - If CCR1[SAMPLE_EN] = 1, the external SAMPLE input is used as the sampling clock.
  - If CCR1[SAMPLE_EN] = 0, the divided bus clock is used as the sampling clock.
- The filter block can be bypassed when not in use.

```
Bypass_Filter_Block = (FILT_CNT == 0x00) |  (~SAMPLE_EN & (FILT_PER == 0x00))
```

- Both COUTA and COUT can be configured as module output CMPO by configuring the CCR1[COUT_SEL] bit and are used for different purposes within the system.
- The optionally filtered COUT can be read directly through register bit CSR[COUT].
- The SAMPLE/WINDOW signal can be inverted by setting CCR1[WINDOW_INV].
- The SAMPLE/WINDOW signal can be closed by CMPO's falling edge or rising edge or both edge by setting CCR1[WINDOW_CLS] in window mode.
- In the window mode, when window is closed, COUTA value can be defined as CCR1[COUTA_OW] by setting CCR1[COUTA_OWEN]. If CCR1[COUTA_OWEN] is not set, COUTA holds the last sampled value when window is closed

NOTE

See the chip configuration section for the source of SAMPLE/WINDOW input.

58.3.2 Functional Modes

The comparator window and filter features can be combined as shown in the following table. Individual modes are discussed below the table.

Table 280. Functional modes

Mode #	CMP_EN	WINDOW_EN	SAMPLE_EN	FILT_CNT	FILT_PER	Operation	Maximum latency[1]
1	0	X	X	X	X	See the Disabled Mode (#1).	N/A
2A	1	0	X	0x00	X	See the Continuous Mode (#2A & 2B).	T_{PD}
2B	1	0	0	X	0x00		
3A	1	0	1	0x01	X	See the Sampled, Non-Filtered Mode (#3A & 3B).	$T_{PD} + T_{SAMPLE} + 3T_{per}$
3B	1	0	0	0x01	> 0x00		$T_{PD} + (FILT_PER * T_{per}) + 3T_{per}$
4A	1	0	1	> 0x01	X	See the Sampled, Filtered Mode (#4A & 4B).	$T_{PD} + (FILT_CNT * T_{SAMPLE}) + 3T_{per}$

Table continues on the next page...

Table 280. Functional modes (continued)

Mode #	CMP_EN	WINDOW_EN	SAMPLE_EN	FILT_CNT	FILT_PER	Operation	Maximum latency[1]
4B	1	0	0	> 0x01	> 0x00		T_{PD} + (FILT_CNT * FILT_PER x T_{per}) + 3T_{per}
5A	1	1	0	0x00	X	See the Windowed Mode (#5A & 5B).	T_{PD} + 2T_{per}
5B	1	1	0	X	0x00		
6	1	1	0	0x01	> 0x00	See the Windowed/ Resampled Mode (#6).	T_{PD} + (FILT_PER * T_{per}) + 3T_{per}
7	1	1	0	> 0x01	> 0x00	See the Windowed/Filtered Mode (#7).	T_{PD} + (FILT_CNT * FILT_PER x T_{per}) + 3T_{per}
All other combinations of CMP_EN, WINDOW_EN, SAMPLE_EN, FILT_CNT, FILT_PER are illegal.							

1. T_{PD} represents the intrinsic delay of the analog component plus the polarity select logic. T_{SAMPLE} is the clock period of the external sample clock. T_{per} is the period of the bus clock.

58.3.2.1 Disabled Mode (#1)

In disabled mode, the analog comparator is non-functional and consumes no power. CSR[COUT] and CMPO is same as CCR1[COUT_INV] in this mode.

58.3.2.2 Continuous Mode (#2A & 2B)

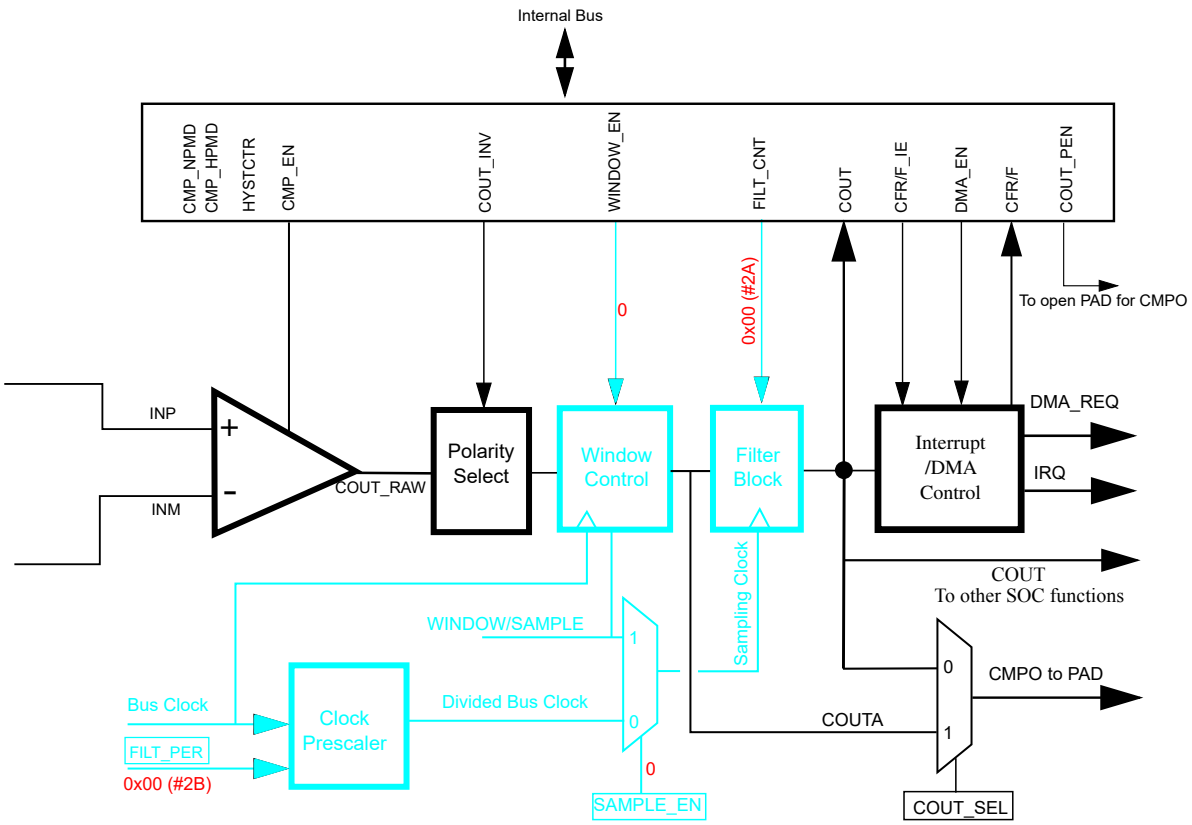


Figure 238. Comparator Operation in Continuous Mode

In this mode, COUT_RAW may be optionally inverted, but is not subject to external sampling or filtering. Both window control and filter blocks are completely bypassed. CSR[COUT] is updated continuously. The path from comparator input pins to output pin is operating in combinational(unclocked) mode. COUT and COUTA are identical.

For cases where a comparator is used to drive a fault input, for example, for a motor-control module such as FTM, it should generally be configured to operate in continuous mode so that an external fault can immediately pass through the comparator to the target fault circuitry.

58.3.2.3 Sampled, Non-Filtered Mode (#3A & 3B)

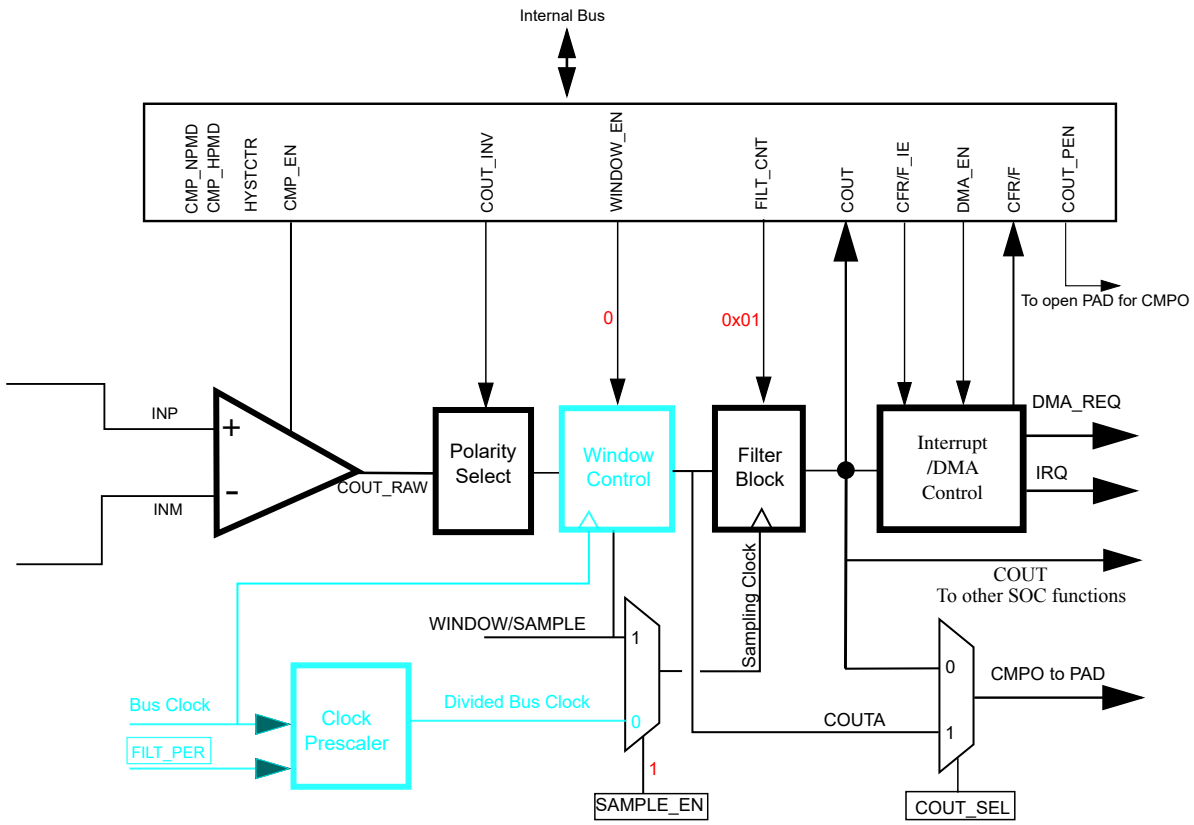


Figure 239. Sampled, Non-Filtered (#3A): Sampling Point Externally Driven

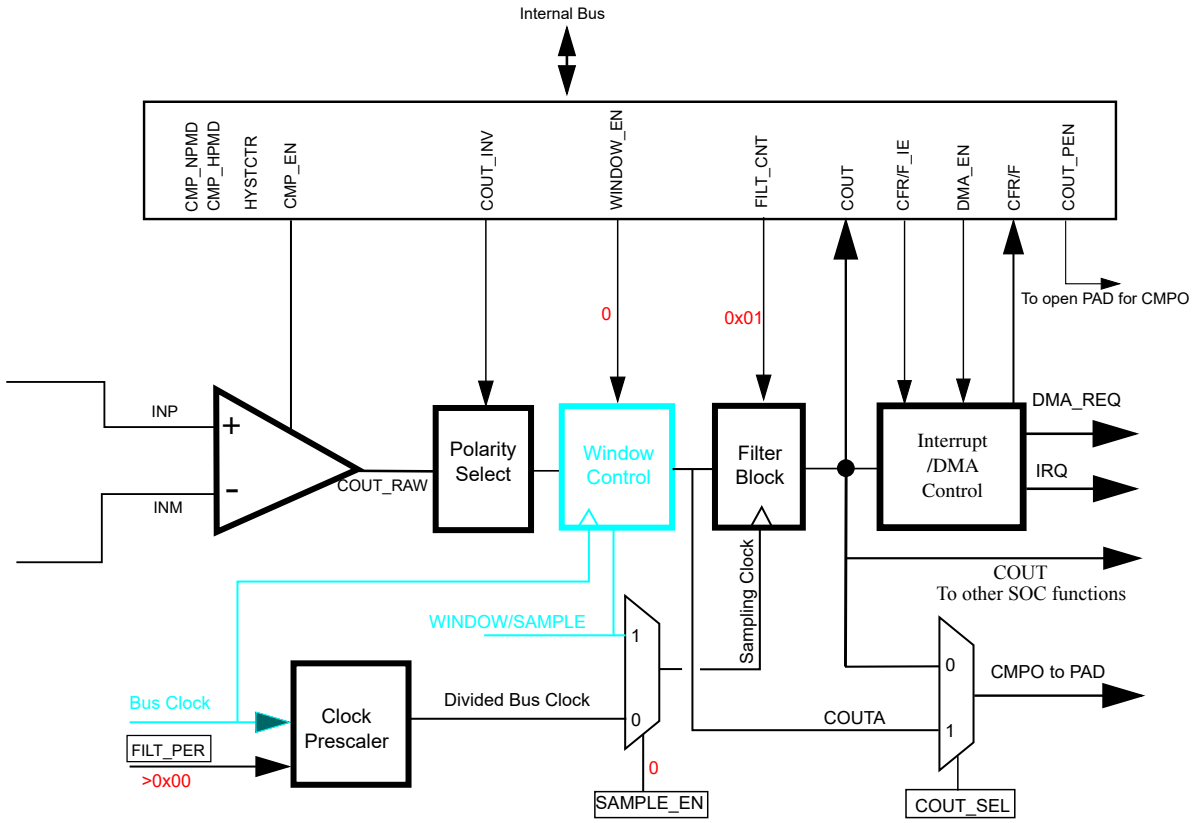


Figure 240. Sampled, Non-Filtered (#3B): Sampling Interval Internally Derived

In this mode, the path from analog inputs to COUTA is combinational(unclocked). Windowing control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the sampling clock.

The only difference in operation between sampled, non-filtered (#3A) and sampled, non-filtered (#3B) is in how the clock to the filter block is derived. In #3A, the clock to filter block is externally derived while in #3B, the clock to filter block is internally derived.

The filter block has no other function than sample/hold of the comparator output in this mode.

The following figure illustrates comparator operation in this mode, assuming that the polarity select is set to non-inverting state.

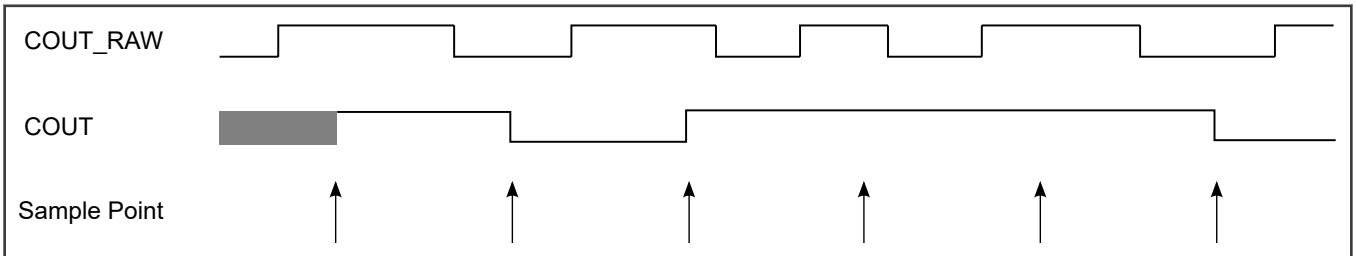


Figure 241. Sampled, Non-Filtered Mode Timing Diagram

58.3.2.4 Sampled, Filtered Mode (#4A & 4B)

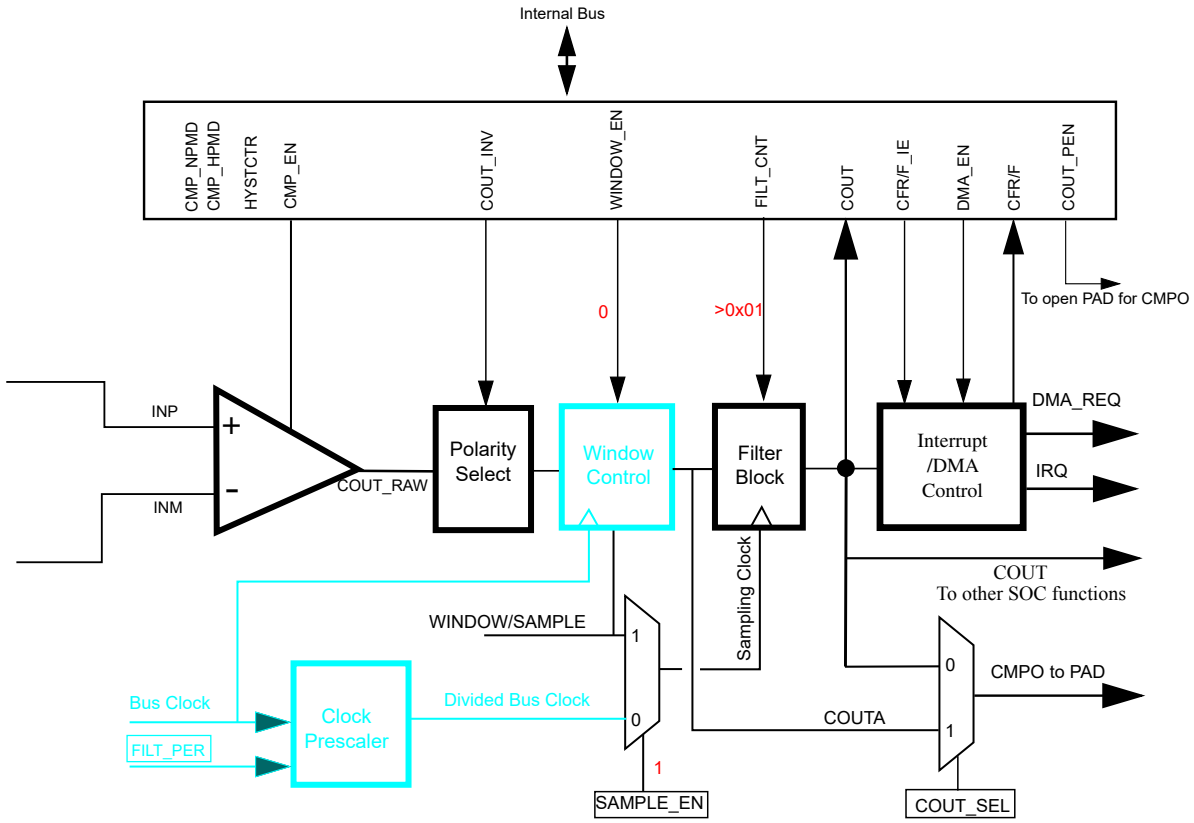


Figure 242. Sampled, Filtered (#4A): Sampling Point Externally Driven

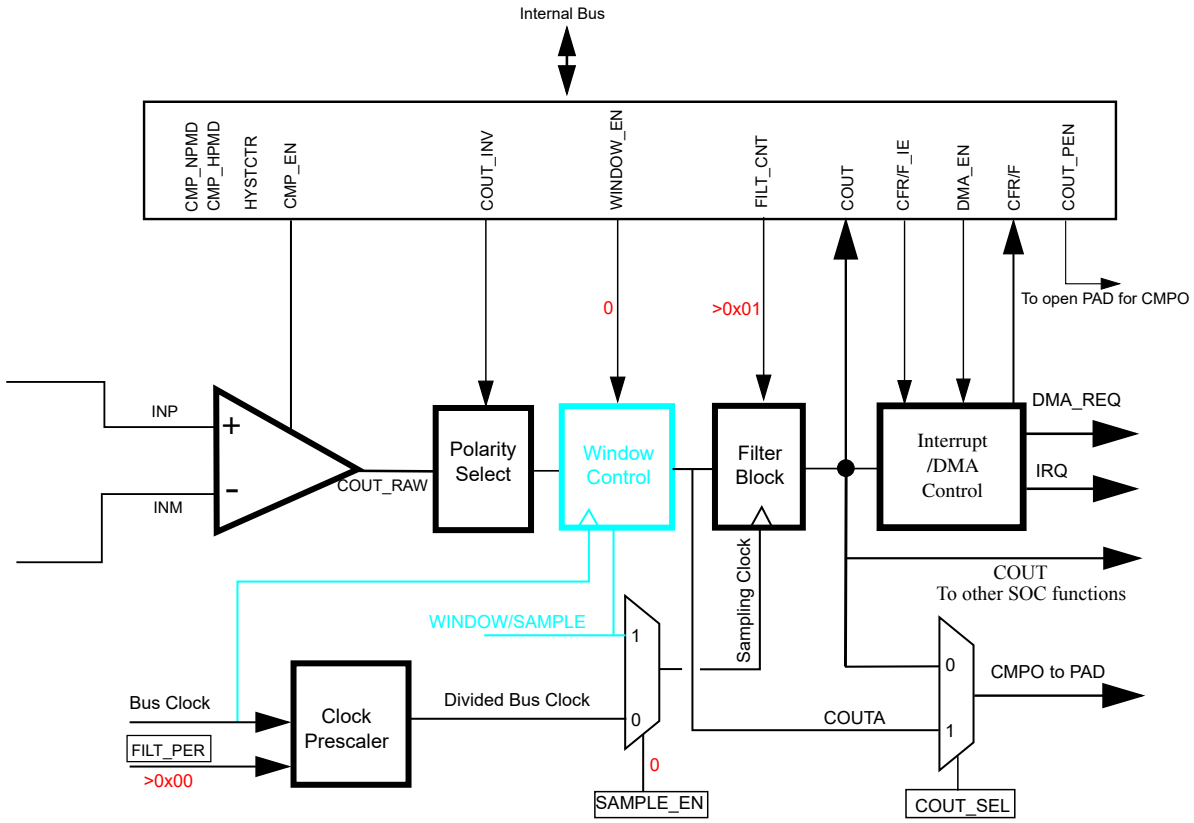


Figure 243. Sampled, Filtered (#4B): Sampling Point Internally Derived

In this mode, the path from analog inputs to COUTA is combinational(unclocked). Windowing control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the sampling clock.

The only difference in operation between sampled, non-filtered (#3A) and sampled, filtered (#4A) is that, now, CCR1[FILT_CNT]>1, which activates filter operation.

The only difference in operation between sampled, non-filtered (#3B) and sampled, filtered (#4B) is that now, CCR1[FILT_CNT]>1, which activates filter operation.

58.3.2.5 Windowed Mode (#5A & 5B)

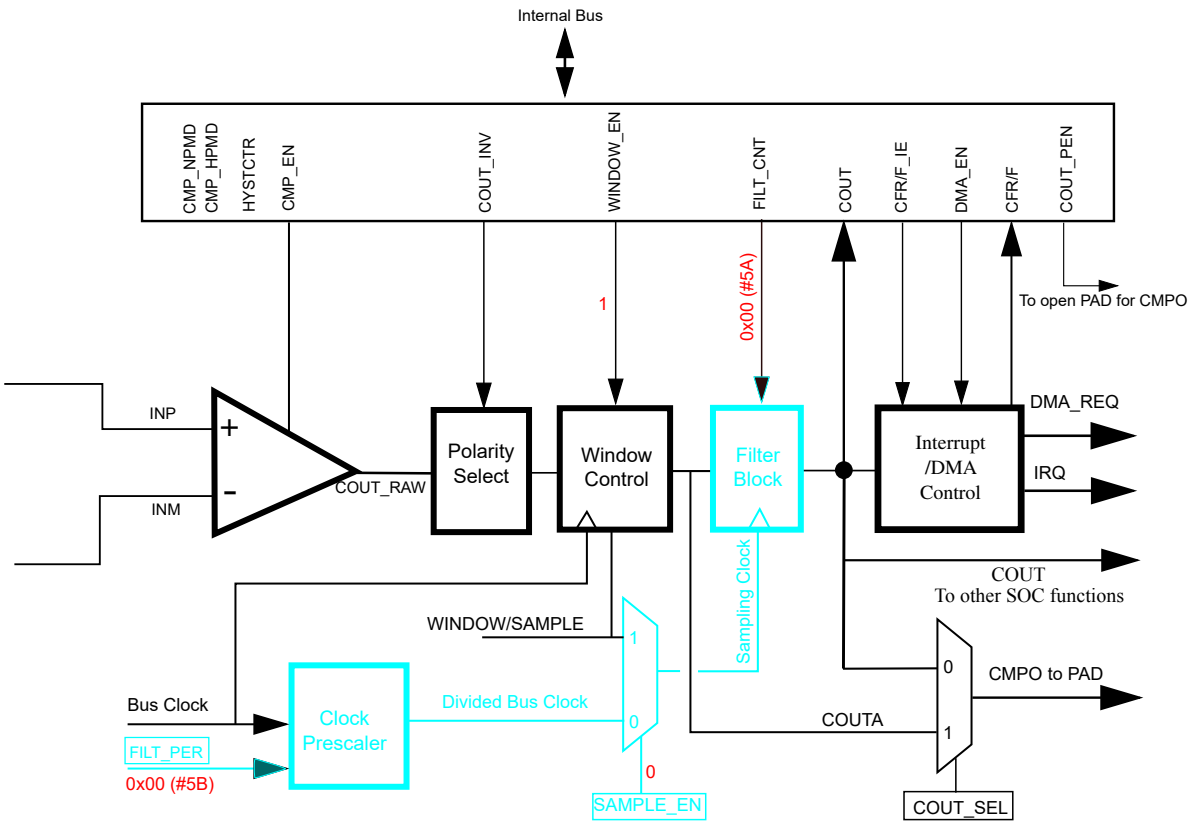


Figure 244. Windowed Mode

In this mode, COUTA is clocked by the bus clock whenever window is enabled. The last latched value is held when window is disabled. The filter block is bypassed.

The following figure illustrates comparator operation in this mode, ignoring latency of the analog comparator, polarity select, and window control block. It also assumes that the polarity select is set to non-inverting state.

In actual operation, COUTA may lag the analog inputs by up to two bus clock cycles plus the combinational path delay through the comparator and polarity select logic.

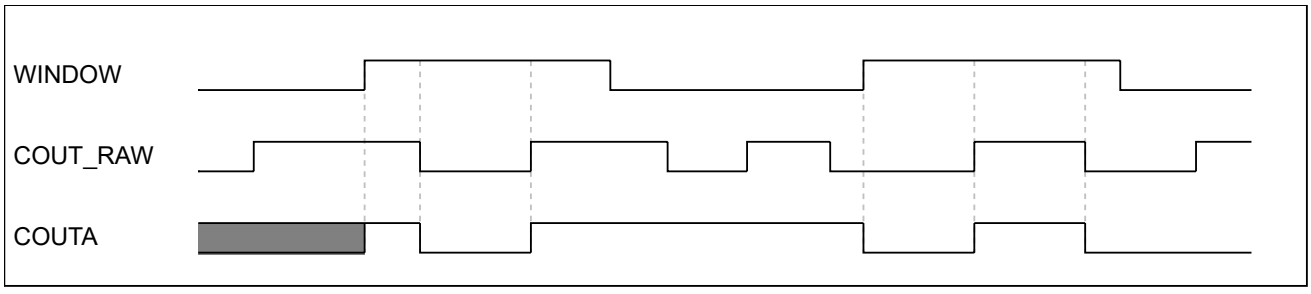


Figure 245. Windowed Mode Timing Diagram

If CCR1[COUTA_OWEN] is set, user can define COUTA level as CCR1[COUTA_OW], when window is closed. Following figures show this case.

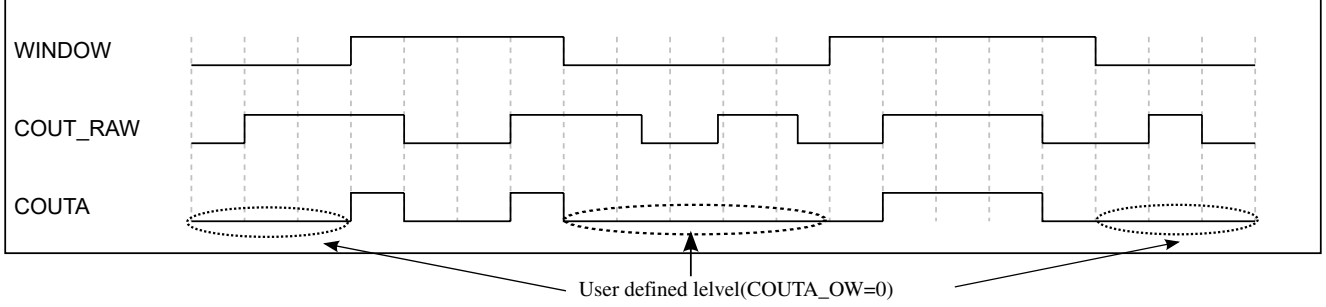


Figure 246. Windowed Mode Timing Diagram With User Defined Value 0 outside WINDOW

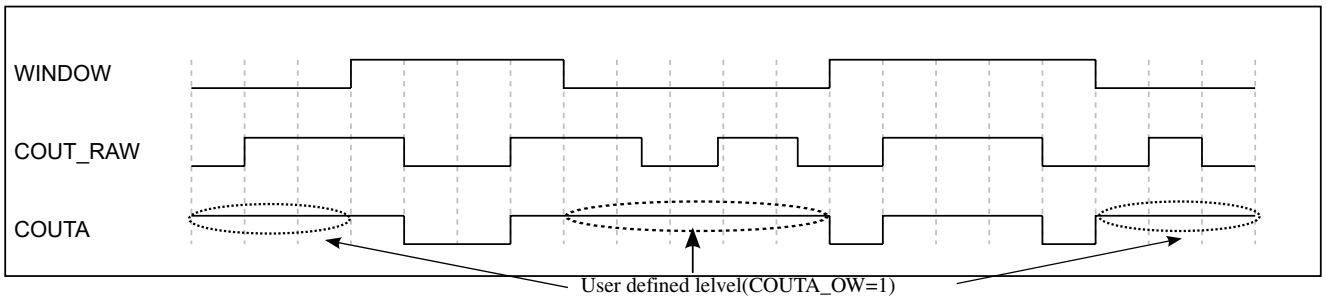


Figure 247. Windowed Mode Timing Diagram With User Defined Value 1 outside WINDOW

NOTE

Obviously, when the window is open, COUT_A will switch from COUTA_OW to COUT_RAW. When the window is closed, COUT_A will switch from COUT_RAW to COUTA_OW. This may generate unnecessary transition flags, CFR or CFF. User needs to choose COUTA_OW carefully according to the actual application, and select the appropriate flag CFR or CFF to generate interrupt.

If CCR1[WINDOW_CLS] is set, user can define the CMPO event (rising, falling or both edges, selected by CCR1[EVT_SEL]) to close the window. The external WINDOW signal has to go to zero and back to one to enable the internal window again. Following figure shows an example that CMPO rising edge close the internal window.

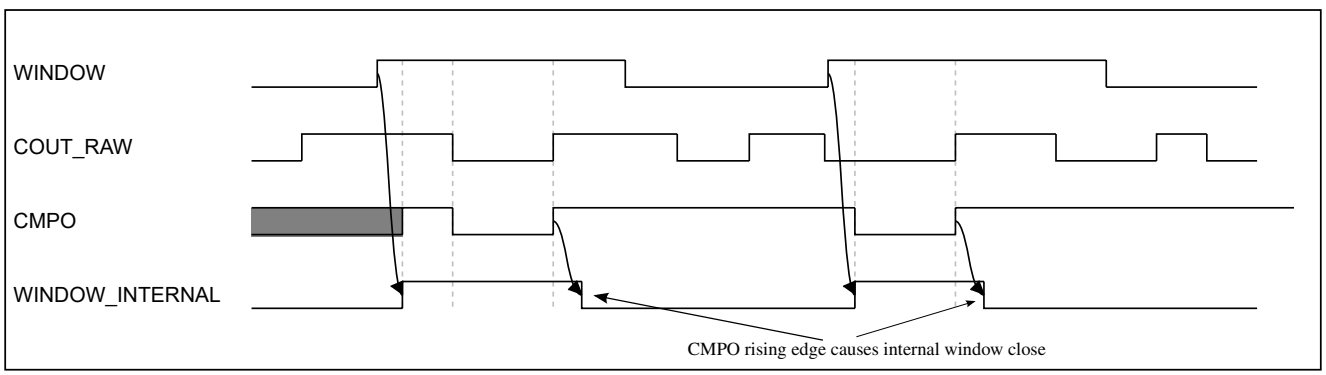


Figure 248. Windowed Mode Timing Diagram With CMPO Rising Edge Close Window

If CCR1[WINDOW_INV] is set, user can invert the window signal before it is used. The following figure shows this case.

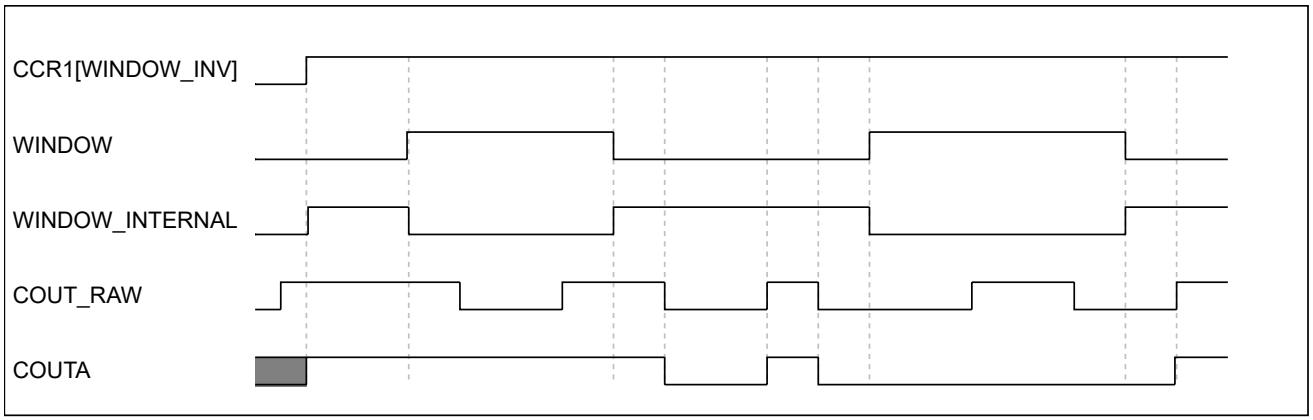


Figure 249. Windowed Mode Timing Diagram With WINDOW Signal Inverted.

58.3.2.6 Windowed/Resampled Mode (#6)

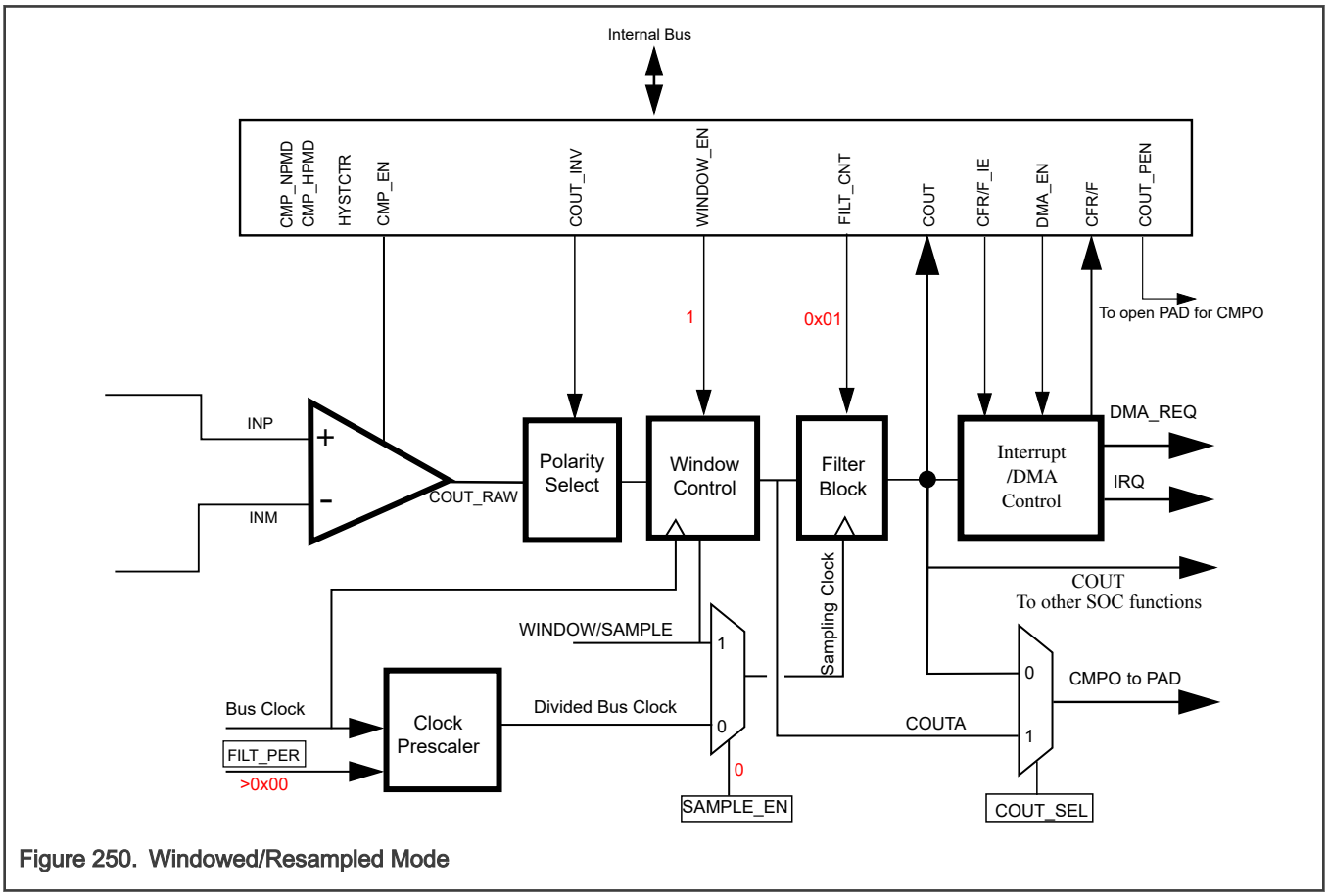


Figure 250. Windowed/Resampled Mode

This mode of operation results in an unfiltered string of comparator samples where the interval between the samples is determined by CCR1[FILT_PER] and the bus clock rate. Configuration for this mode is virtually identical to that for the windowed/filtered mode shown in the next section. The only difference is that the value of CCR1[FILT_CNT] in this mode must be 1.

The following figure uses the same input stimulus shown in Figure 245, and adds resampling of COUTA to generate COUT. Samples are taken at the time points indicated by the arrows in the figure. Again, prop delays and latency are ignored for the sake of clarity.

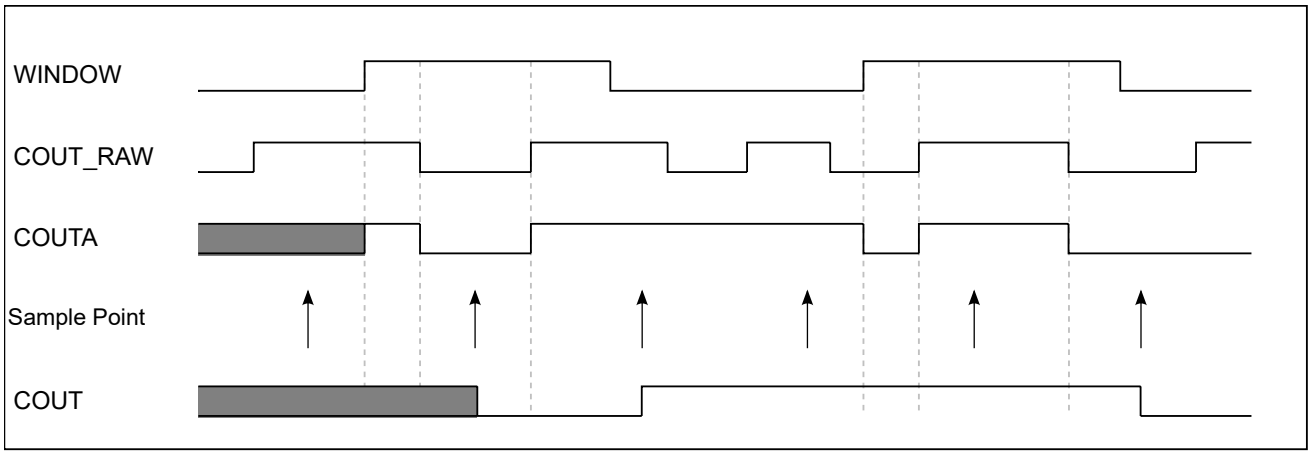


Figure 251. Windowed/Resampled Mode Operation

This example was generated solely to demonstrate operation of the comparator in windowed/resampled mode, and does not reflect any specific application. Depending upon the sampling rate and window placement, COUT may not see zero-crossing events detected by the analog comparator. Sampling period and/or window placement must be carefully considered for a given application.

58.3.2.7 Windowed/Filtered Mode (#7)

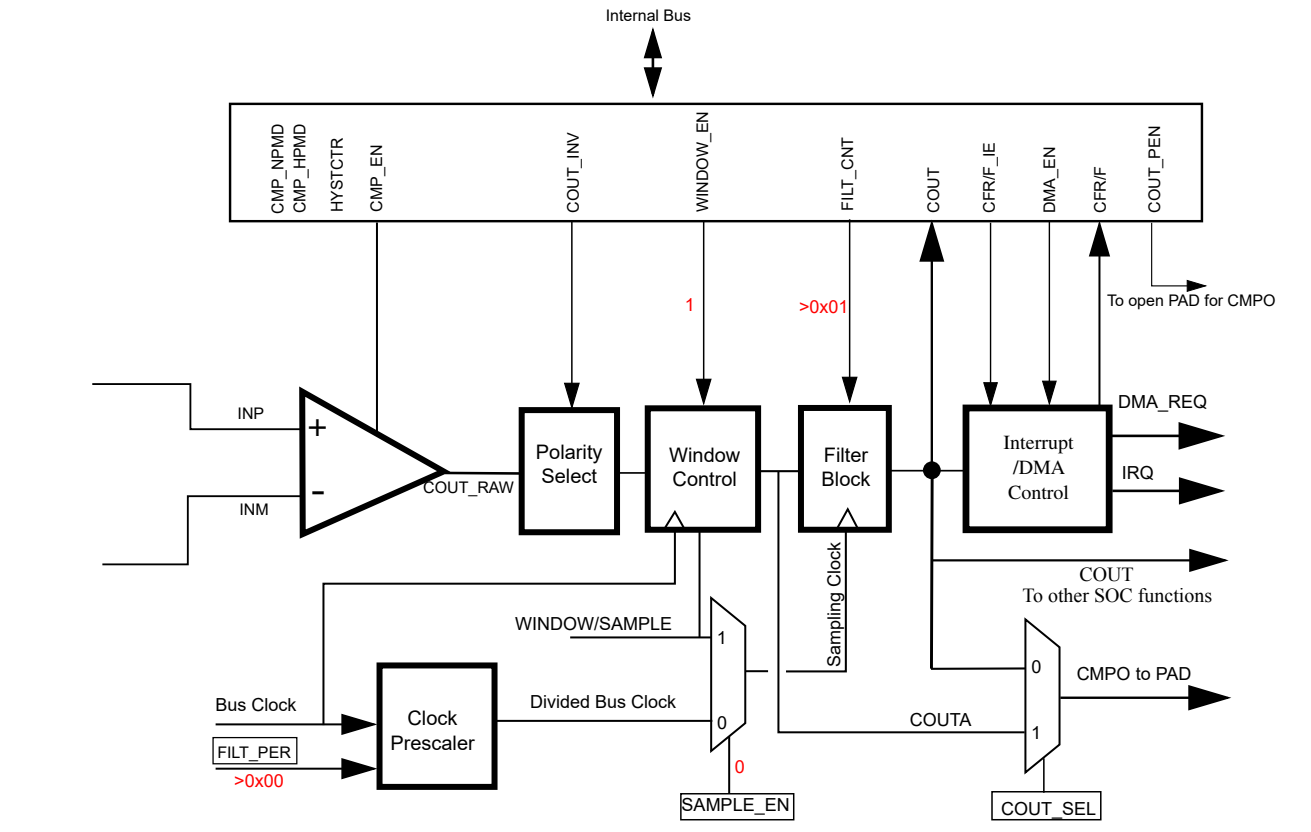


Figure 252. Windowed/Filtered Mode

The only difference in operation between Windowed/Resampled Mode (#6) and Windowed/Filtered Mode (#7) is that, now, CCR1[FILT_CNT]>1, which activates filter operation.

This is the most complex mode of operation for the comparator block, as it utilizes both windowing and filtering features. It also has the highest latency of any of the modes. This can be approximated: up to 2 peripheral clock synchronization in the window function + ((CCR1[FILT_CNT] X CCR1[FILT_PER]) + 1) X peripheral clock for the filter function.

58.3.3 Trigger Mode

The trigger mode is enabled with RRCR0[RR_EN] and CCR0[CMP_EN]. A trigger event will initiate a comparison sequence. The next trigger event should not come before the current sequence completes.

The reference channel for either the plus side MUX or the minus side MUX is selected by RRCR1[RR_FIXP] and RRCR1[FIXCH]. The active channels are selected by RRCR1[RR_CH*n*EN].

When a trigger comes, the analog comparator is enabled. After the comparison sequence completes, the analog comparator is disabled again. The analog stabilization time is controlled by RRCR0[RR_INITMOD]. Make sure that the (RR_INITMOD*round robin clock period) should be longer than the initialization delay in the chip data sheet.

When the stabilization process completes, the round robin manner comparison sequence begins. After the configurable number of operation clocks defined by RRCR0[RR_NSAM], the comparison result is sampled for the selected active channel.

After all the active channels are sampled/compared, if the comparison result changes from its pre-programmed state, the corresponding flag in RRSR[RR_CH*n*F] is set. The pre-programmed state for each channel is configured by writing to RRCSR[RR_CH*n*OUT] field. RRCSR[RR_CH*n*OUT] is updated to store the last comparison result for each channel. If any flag in RRSR[RR_CH*n*F] is set, flag CSR[RRF] will be set. If IER[RRF_IE] is set, an asynchronous interrupt is asserted. Note that these flags do not support generating a DMA transfer event.

The following diagram shows the basic flow of this mode. In the diagram, RRCR1[RR_CH1EN], RRCR1[RR_CH3EN], and RRCR1[RR_CH4EN] are set, so channels #1, #3, and #4 are selected for round robin depended on their priority setting. RRCR0[RR_NSAM] is set to 2'b01, so one clock later the comparison result of the selected channel is sampled. When channel #4 is compared, the result is sampled, and round robin ends. If any of the comparison results from channel #1, #3, or #4 changed from their programmed value (written to RRCSR[RR_CH1OUT], RRCSR[RR_CH3OUT], and RRCSR[RR_CH4OUT]), an interrupt is generated. Software can then poll the RRSR[RR_CH*n*F] to see which channel input(s) changed value.

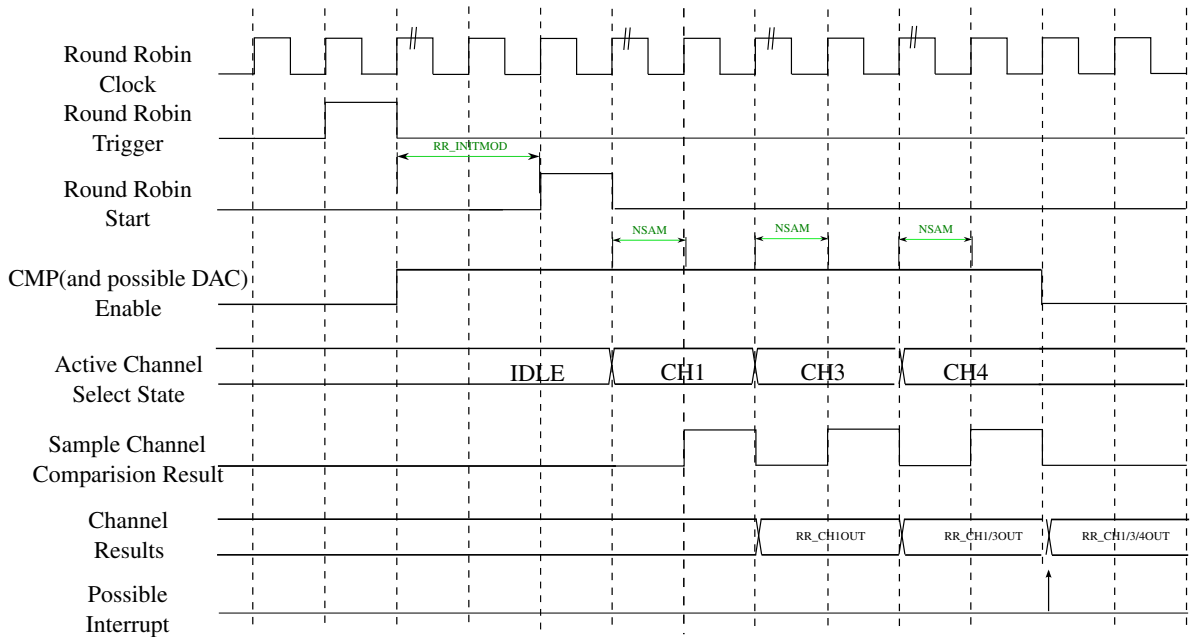


Figure 253. Trigger Mode

Below table shows the channel decode in both functional mode and trigger mode. Other cases not in the table are illegal.

Table 281. CMP Channel Decode in Functional Mode and Trigger mode

Mode	RR_EN	PSEL[2:0]	MSEL[2:0]	INPSEL[1:0]	INMSEL[1:0]	FIXP	FIXCH[2:0]	RR_CHxN	INP	INM	CMP Behavior
Functional Mode	0	x[1]	0~7	0	1	x	x	x	DAC	Channel decoded from MSEL[2:0]	Channel 0~7 can be compared with DAC
		0~7	x	1	0	x	x	x	Channel decoded from PSEL[2:0]	DAC	Channel 0~7 can be compared with DAC
		0~7	0~7	1	1	x	x	x	Channel decoded from PSEL[2:0]	Channel decoded from MSEL[2:0]	Channel 0~7 can be compared with channel 0~7[2]
Trigger Mode	1	x	x	0	1	0	x	0~7	DAC	Channel sweep (RR_CHxN)	Channel 0~7 can be swept with DAC
		x	x	1	0	1	x	0~7	Channel sweep (RR_CHxN)	DAC	Channel 0~7 can be swept with DAC
		x	x	1	1	0	0~7	0~7	Channel fixed by FIXCH[2:0]	Channel sweep (RR_CHxN)	Channel 0~7 can be swept with a fixed channel(0~7)[3]
		x	x	1	1	1	0~7	0~7	Channel sweep (RR_CHxN)	Channel fixed by FIXCH[2:0]	Channel 0~7 can be swept with a fixed channel(0~7)[3]

1. "x" means "do not care"
2. PSEL should not be same as MSEL.
3. Channel in the sweep side should not be same as the fixed side.

Trigger Mode Programming Recommendation

It is recommended to configure the Trigger Mode as following steps. Register fields in same register can be configured at one time.

1. Configure the comparison cycles by RRCR0[RR_NSAM]. Note: It is a mandatory request that the round robin cycling period must be set longer than the time that all the active channels complete the specified comparison cycles set by RRCR0[RR_NSAM].
2. Configure CMP initialization delay by RRCR0[RR_INITMOD] Note: In programming the RRCR0[RR_INITMOD] registers, the RR_INITMOD x round robin clock period must be longer than the initialization delay, which can be referred from the chip data sheet.
3. Configure the RRCR1[FIXP] to select the fixed port of CMP and
    a. If using one input channel to compare with other channels, configure the RRCR1[FIXCH] to select the fixed channel
    b. If using DAC output to compare with input channels, configure the CCR2[INPSEL] or CCR2[INMSEL](according to RRCR1[FIXP])to select the DAC output
4. Configure channels for comparison by RRCR1[RR_CHnEN]
5. Write RRCSR[RR_CHnOUT] to define the pre-set state of channel n.
6. Clear channel flags RRSR[RR_CHnF]
7. Enable round robin interrupt by IER[RRF_IE] (disable IER[CFR_IE] and IER[CFF_IE])
8. Enable round robin mode by RRCR0[RR_EN]
9. Enable comparator by CCR0[CMP_EN]

58.3.4 Low-Pass Filter

The low-pass filter operates on the unfiltered, optionally inverted comparator output COUTA and generates the filtered and synchronized output COUT. Both COUTA and COUT can be configured as module outputs and are used for different purposes within the system.

Synchronization and edge detection are always used to determine status register bit values. They also apply to COUT for all sampling and windowed modes. Filtering can be performed using an internal timebase defined by CCR1[FILT_PER], or using an external SAMPLE input to determine sample time.

The need for digital filtering and the amount of filtering is dependent on user requirements. Filtering can become more useful in the absence of an external hysteresis circuit. Without external hysteresis, high-frequency oscillations can be generated at COUTA when the selected INM and INP input voltages differ by less than the offset voltage of the differential comparator.

58.3.4.1 Enabling Filter Modes

Filter modes can be enabled by:

- Setting CCR1[FILT_CNT] > 0x01 and
- Setting CCR1[FILT_PER] to a nonzero value or setting CCR1[SAMPLE_EN]=1

If using the divided bus clock to drive the filter, it samples COUTA every CCR1[FILT_PER] bus clock cycles.

If CCR1[SAMPLE_EN]=1, the filter samples COUTA on each positive transition of the SAMPLE input. The output state of the filter changes when all the consecutive CCR1[FILT_CNT] samples agree that the output value has changed.

58.3.4.2 Latency Issues

The value of CCR1[FILT_PER] or SAMPLE period must be set such that the sampling period is just longer than the period of the expected noise. This way a noise spike will corrupt only one sample. The value of CCR1[FILT_CNT] must be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of CCR1[FILT_CNT].

The values of CCR1[FILT_PER] or SAMPLE period and CCR1[FILT_CNT] must also be traded off against the desire for minimal latency in recognizing actual comparator output transitions. The probability of detecting an actual output change within the nominal latency is the probability of a correct sample raised to the power of CCR1[FILT_CNT].

Table 280 summarizes maximum latency values for the various modes of operation in the absence of noise. Filtering latency is restarted each time an actual output transition is masked by noise.

58.3.5 Low Power Mode Operation

The following table shows the STANDBY mode operation of the LPCMP module.

Table 282. Low Power Mode Operation

Mode of Operation	Description
STANDBY	LPCMP can operate only in continuous mode or trigger mode.

58.3.6 DMA

When DMA support is enabled by setting CCR1[DMA_EN] and the interrupt is enabled by setting IER[CFR_IE], IER[CFF_IE], or both, the corresponding change on COUT forces a DMA transfer request rather than a CPU interrupt instead. When the DMA has completed the transfer, it sends a transfer completing indicator signal that de-asserts the DMA transfer request and clears the flags (both CSR[CFR] and CSR[CFF]) to allow a subsequent change on comparator output to occur and force another DMA request.

58.3.7 Clocks

LPCMP requires the following clocks to operate:

- Bus clock is used for register access and window/filter function.
- Round robin clock for trigger mode.

58.3.8 Resets

The global chip reset signal resets LPCMP.

58.3.9 Interrupts

When corresponding interrupt enable register bit is set, flags CSR[CFR], CSR[CFF], CSR[RRF] can generate interrupt, assuming the DMA enable bit is not set. The interrupt is de-asserted by clearing either the flag or the interrupt enable register bit.

58.4 External Signals

Table 283. External Signal Descriptions

Signal	Description	I/O
CMPO	Filtered or unfiltered comparator output	O
Input_Analog_Channels	Analog input channels(Refer to the chip-specific information for the connections)	I
VREFH_EXT	External reference voltage for the CMP-DAC(Refer to the chip-specific information for the connections)	I

58.5 Initialization

To enable LPCMP, configure the control registers(CCR1, CCR2, DCR, etc) before setting the CCR0[CMP_EN] bit. To disable the module, clear the CCR0[CMP_EN] bit. Switching operation modes or changing control register bits on the fly (when

CCR0[CMP_EN] is 1) may cause noise on the COUT or COUTA signals. To avoid unwanted signal noise, be sure to disable the module before switching modes or changing control bits.

The time required to stabilize COUT is the power-on delay of the comparators plus the largest propagation delay from a selected analog source through the analog comparator, windowing function, and filter. See the data sheet for power-on delays of the comparators. The delay caused by windowing and filter function is specified in the Table 280.

During operation, the propagation delay of the selected data paths must always be considered. It can take many bus clock cycles for COUT and CSR[CFR]/CSR[CFF] to reflect an input change or a configuration change to one of the components involved in the data path.

58.6 Application Information

58.6.1 Round-robin clock(RCLK) frequency requirement

(1) RCLK High Frequency Limit

RCLK high frequency limit depends on two facts:

1. The analog CMP and DAC initialization time(See the chip data sheet for the initialization time.)
   - Register field RR_INITMOD provides a maximum 63 RCLK cycles for the analog CMP and DAC initialization.
   - Rclk must be slow enough to satisfy: 63 * $(1/f_{RCLK}) > T_{inialization}$, where f_{RCLK} is in MHz, and $T_{inialization}$is in microsecond.
   - so $f_{RCLK} < 63 / T_{inialization}$
   - Example: $T_{inialization}$ = 40 microsecond, then f_{RCLK} should be smaller than 1.575MHz.
2. The analog CMP propagation delay(See the chip datasheet for the CMP propagation delay.)
   - Register field NSAM field provides a maximum 4 RCLK cycles for the analog CMP propagation delay.
   - Rclk must be slow enough to satisfy: 4 * $(1/f_{RCLK}) > T_{propagation}$, where f_{RCLK} is in MHz, and $T_{propagation}$ is in microsecond.
   - so $f_{RCLK} < 4 / T_{propagation}$
   - Example: $T_{propagation}$ = 0.1 microsecond, then f_{RCLK} should be smaller than 40 MHz.

(2) RCLK Low Frequency Limit

In theory, RCLK frequency has no low limit. But the lower the RCLK frequency, the longer the scan time. Therefore, the lower limit of the RCLK frequency depends on the system application.

58.7 LPCMP register descriptions

The memory map comprises of 32-bit aligned registers which can be accessed via 8-, 16- or 32-bit reads and 32-bit write. Attempted accesses using unsupported write data sizes, writes to read-only resources, or to reserved spaces are terminated with an error. Read access to reserved address will also generate a transfer error and the read data bus will show all 0s.

58.7.1 LPCMP memory map

LPCMP_0 base address: 4037_0000h

LPCMP_1 base address: 4037_4000h

LPCMP_2 base address: 404E_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	R	0100_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4h	Parameter Register (PARAM)	32	R	0000_0002h
8h	Comparator Control Register 0 (CCR0)	32	RW	0000_0002h
Ch	Comparator Control Register 1 (CCR1)	32	RW	0000_0000h
10h	Comparator Control Register 2 (CCR2)	32	RW	0000_0000h
18h	DAC Control Register (DCR)	32	RW	0000_0000h
1Ch	Interrupt Enable Register (IER)	32	RW	0000_0000h
20h	Comparator Status Register (CSR)	32	RW	0000_0000h
24h	Round Robin Control Register 0 (RRCR0)	32	RW	0000_0000h
28h	Round Robin Control Register 1 (RRCR1)	32	RW	0000_0000h
2Ch	Round Robin Control and Status Register (RRCSR)	32	RW	0000_0000h
30h	Round Robin Status Register (RRSR)	32	RW	0000_0000h

58.7.2 Version ID Register (VERID)

Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram

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

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

Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module design.
23-16 MINOR	Minor Version Number Returns the minor version number for the module design.
15-0 FEATURE	Feature Specification Number Returns the feature set number. 0000_0000_0000_0001b - Round robin feature

58.7.3 Parameter Register (PARAM)

Offset

Register	Offset
PARAM	4h

Function
Contains parameter values that are implemented in the module.

Diagram

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

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

Fields

Field	Function
31-4 —	Reserved
3-0	DAC Resolution

Table continues on the next page...

Table continued from the previous page...

Field	Function
DAC_RES	Indicates the DAC resolution supported by this implementation. 0000b - 4 bit DAC 0001b - 6 bit DAC 0010b - 8 bit DAC 0011b - 10 bit DAC 0100b - 12 bit DAC 0101b - 14 bit DAC 0110b - 16 bit DAC

58.7.4 Comparator Control Register 0 (CCR0)

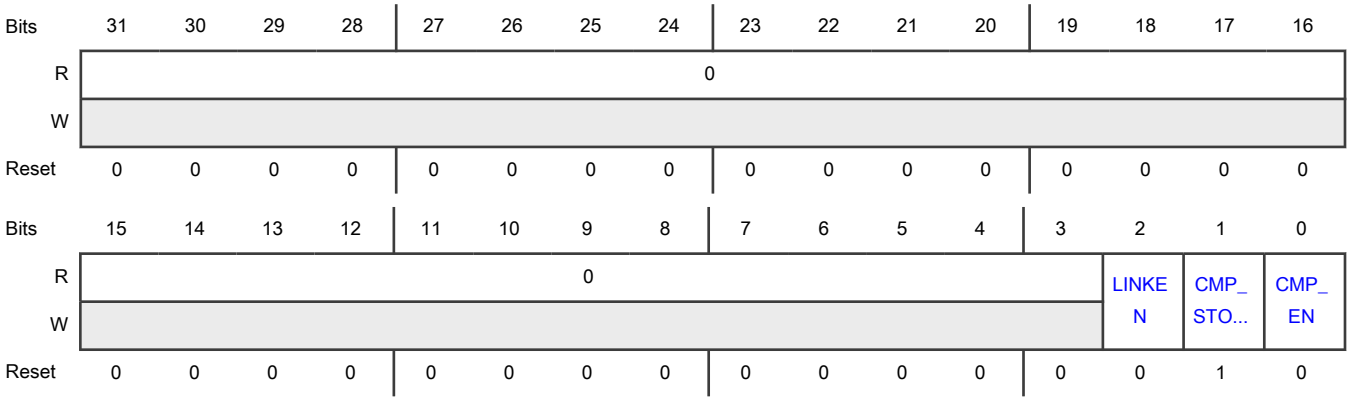
Offset

Register	Offset
CCR0	8h

Function

Contains configuration options for enabling the analog comparator and the DAC.

Diagram



Fields

Field	Function
31-3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 LINKEN	CMP-to-DAC Link Enable Controls the link from the CMP enable to the DAC enable. 0b - Disable the CMP-to-DAC link: enabling or disabling the DAC is independent from enabling or disabling the CMP. 1b - Enable the CMP-to-DAC link: the DAC enable/disable is controlled by the CMP_EN bit instead of DCR[DAC_EN]. Also, when the CMP is auto-disabled because software selects the same signal for both the plus and minus comparator inputs, the DAC is disabled too.
1 CMP_STOP_EN	Comparator STANDBY Mode Enable Allows software to enable the analog comparator or the DAC when the device is in STANDBY mode. **NOTE** CMP_STOP_EN has no effect in trigger mode. 0b - Disable the analog comparator regardless of CMP_EN. 1b - Allow the analog comparator to be enabled by CMP_EN.
0 CMP_EN	Comparator Enable Enables the analog comparator. **NOTE** When CCR0[LINKEN]=1, CMP_EN also controls the enabling/disabling of the DAC instead of DCR[DAC_EN]. 0b - Disable (The analog logic remains off and consumes no power.) 1b - Enable

58.7.5 Comparator Control Register 1 (CCR1)

Offset

Register	Offset
CCR1	Ch

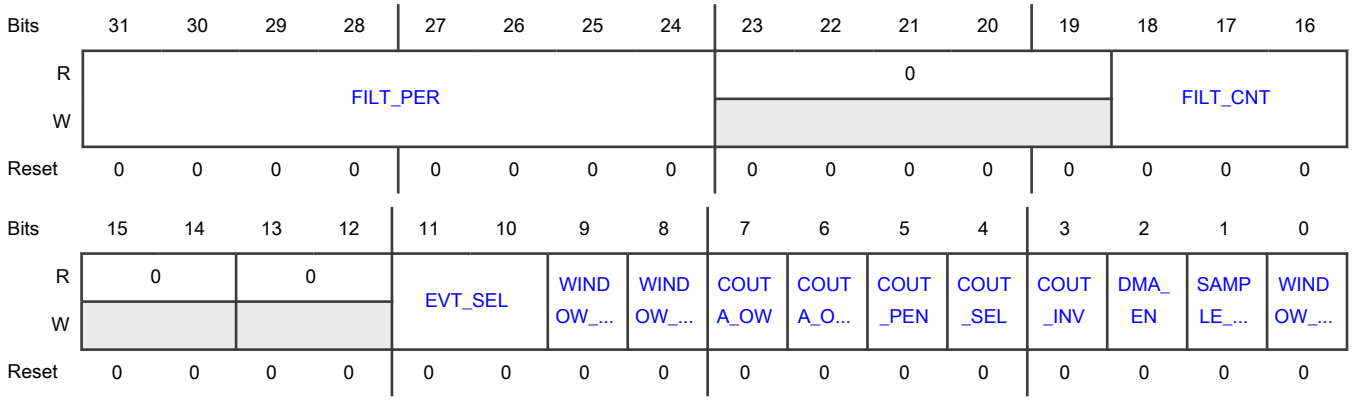
Function

Contains configuration options for comparator operation, such as enabling Sampling or Windowing mode.

NOTE

Sampling and Windowing modes cannot both be enabled at the same time. Sampling mode takes precedence over Windowing mode. If software attempts to set both SAMPLE_EN and WINDOW_EN, only SAMPLE_EN is set.

Diagram



Fields

Field	Function
31-24 FILT_PER	Filter Sample Period Specifies the sampling period (in bus clock cycles) of the comparator output filter. Programming FILT_PER to 0x00 bypasses the filter. Filter programming and latency details are provided in the functional description section. **NOTE** FILT_PER has no effect in Sampling mode (CCR1[SAMPLE_EN]=1).
23-19 —	Reserved
18-16 FILT_CNT	Filter Sample Count Specifies the number of consecutive samples that must agree before the comparator ouput filter accepts the sample as a new valid output state. For information regarding filter programming and latency, see the functional description section. 000b - Filter is bypassed: COUT = COUTA 001b - 1 consecutive sample (Comparator output is simply sampled.) 010b - 2 consecutive samples 011b - 3 consecutive samples 100b - 4 consecutive samples 101b - 5 consecutive samples 110b - 6 consecutive samples 111b - 7 consecutive samples
15-14 —	Reserved
13-12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11-10 EVT_SEL	CMPO Event Select Selects which CMPO signal edge (rising, falling, or both) defines a CMPO event. **NOTE** Valid only in Windowing mode. 00b - Rising edge 01b - Falling edge 1xb - Both edges
9 WINDOW_CLS	CMPO Event Window Close Enables a CMPO event (defined as a CMPO rising edge, falling edge, or both) to close an active window. See the EVT_SEL field to configure the CMPO event. **NOTE** The WINDOW signal has to go to zero and back to one again to re-activate the window. Valid only in Windowing mode. 0b - CMPO event cannot close the window 1b - CMPO event can close the window
8 WINDOW_INV	WINDOW/SAMPLE Signal Invert Inverts the WINDOW/SAMPLE signal. 0b - Do not invert 1b - Invert
7 COUTA_OW	COUTA Output Level for Closed Window Defines the COUTA signal value when the window is closed. **NOTE** Valid only in Windowing mode and when COUTA_OWEN=1. 0b - COUTA is 0 1b - COUTA is 1
6 COUTA_OWEN	COUTA_OW Enable Enables the COUTA signal value to be defined by the COUTA_OW bit when the window is closed. **NOTE** Valid only in Windowing mode. 0b - COUTA holds the last sampled value 1b - COUTA is defined by the COUTA_OW bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 COUT_PEN	Comparator Output Pin Enable Enables the comparator output to become an available signal option for a selected package pin. 0b - Not available 1b - Available
4 COUT_SEL	Comparator Output Select Selects which comparator output option, COUT or COUTA, to use for CMPO. 0b - Use COUT (filtered) 1b - Use COUTA (unfiltered)
3 COUT_INV	Comparator Invert Selects the polarity of the analog comparator function, affecting the value driven to the COUT output (on both the device pin and as CSR[COUT]) when CCR0[CMP_EN] is 0. **NOTE** COUT_INV has no effect in trigger mode. 0b - Do not invert 1b - Invert
2 DMA_EN	DMA Enable Enables DMA transfers triggered from the LPCMP module. When DMA_EN and the corresponding interrupt enable bit are set, a DMA request is asserted when CFR or CFF is set. 0b - Disable 1b - Enable
1 SAMPLE_EN	Sampling Enable Enables Sampling mode. 0b - Disable 1b - Enable
0 WINDOW_EN	Windowing Enable Enables Windowing mode. **NOTE** Valid only when SAMPLE_EN=0. 0b - Disable 1b - Enable

58.7.6 Comparator Control Register 2 (CCR2)

Offset

Register	Offset
CCR2	10h

Function

Contains configuration options for comparator operation, such as selecting the plus and minus comparator inputs and the hysteresis levels.

NOTE

When an inappropriate operation selects the same signal for both the plus and minus comparator inputs, the analog comparator automatically shuts down (regardless of CMP_EN) to prevent itself from becoming a noise generator.

Diagram

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

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

Fields

Field	Function
31-30 —	Reserved
29-28 INMSEL	Input Minus Select Selects the minus input of the comparator. **NOTE** These selections connect directly to the minus input of the comparator. 00b - IN0: from the 8-bit DAC output 01b - IN1: from the analog 8-1 mux 10b - Reserved 11b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-26 —	Reserved
25-24 INPSEL	Input Plus Select Selects the plus input of the comparator. **NOTE** These selections connect directly to the plus input of the comparator. 00b - IN0: from the 8-bit DAC output 01b - IN1: from the analog 8-1 mux 10b - Reserved 11b - Reserved
23 —	Reserved
22-20 MSEL	Minus Input MUX Select Selects the input used for the negative mux. See the chip-specific LPCMP information to get the detailed connections. **NOTE** MSEL has no effect in trigger mode. 000b - Input 0 001b - Input 1 010b - Input 2 011b - Input 3 100b - Input 4 101b - Input 5 110b - Input 6 111b - Input 7
19 —	Reserved
18-16 PSEL	Plus Input MUX Select Selects the input used for the positive mux. See the chip-specific LPCMP information to get the detailed connections. **NOTE** PSEL has no effect in trigger mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - Input 0 001b - Input 1 010b - Input 2 011b - Input 3 100b - Input 4 101b - Input 5 110b - Input 6 111b - Input 7
15-6 —	Reserved
5-4 HYSTCTR	Comparator Hysteresis Control Selects the level of internally generated hysteresis for the comparator output. See the chip data sheet to get the specific values for each hysteresis level. 00b - Level 0 01b - Level 1 10b - Level 2 11b - Level 3
3 —	Reserved
2 OFFSET	Comparator Offset Control Selects the level of internally generated voltage offset for the comparator output. See the chip data sheet to get the specific values for each offset level. 0b - Level 0: The hysteresis selected by HYSTCTR is valid for both directions (rising and falling). 1b - Level 1: Hysteresis does not apply when INP (input-plus) crosses INM (input-minus) in the rising direction or when INM crosses INP in the falling direction. Hysteresis still applies for INP crossing INM in the falling direction.
1 —	Reserved
0 CMP_HPMD	CMP High Power Mode Select Selects Low or High Power(Speed) mode for the comparator. 0b - Low power(speed) comparison mode 1b - High power(speed) comparison mode

58.7.7 DAC Control Register (DCR)

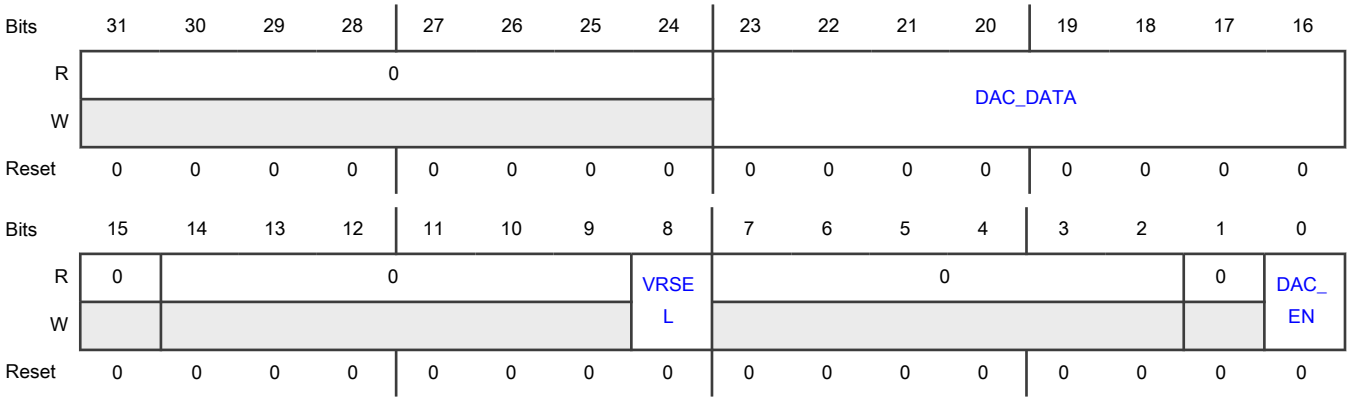
Offset

Register	Offset
DCR	18h

Function

Contains configuration options for enabling the DAC.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 DAC_DATA	DAC Output Voltage Select Selects the DAC output (DACO) voltage from one of 256 distinct levels. The DACO range is from Vin/256 to Vin: DACO = (Vin/256) * (DAC_DATA + 1)
15 —	Reserved
14-9 —	Reserved
8 VRSEL	DAC Reference High Voltage Source Select Selects the high voltage reference source for the Vin supply of the DAC's resistor ladder network. See the chip-specific LPCMP information for the source of vrefh0 and vrefh1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - vrefh0 1b - vrefh1
7-2 —	Reserved
1 —	Reserved
0 DAC_EN	DAC Enable Enables the DAC. When disabled, the DAC is powered down to conserve power. 0b - Disable 1b - Enable

58.7.8 Interrupt Enable Register (IER)

Offset

Register	Offset
IER	1Ch

Function

Provides enable bits for the comparator and round-robin flag interrupts.

Diagram

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

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

Fields

Field	Function
31-3 —	Reserved
2 RRF_IE	Round-Robin Flag Interrupt Enable Enables the Round-Robin Flag interrupt. 0b - Disable 1b - Enable: Assert an interrupt when the comparison result changes for a given channel.
1 CFF_IE	Comparator Flag Falling Interrupt Enable Enables the Comparator Flag Falling interrupt. 0b - Disable 1b - Enable: Assert an interrupt when CFF is set.
0 CFR_IE	Comparator Flag Rising Interrupt Enable Enables the Comparator Flag Rising interrupt. 0b - Disable 1b - Enable: Assert an interrupt when CFR is set.

58.7.9 Comparator Status Register (CSR)

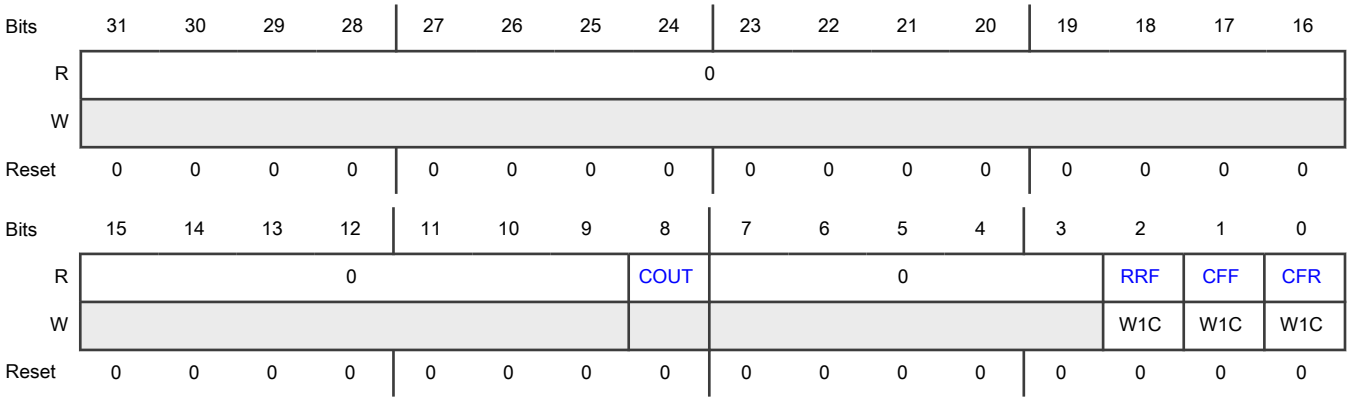
Offset

Register	Offset
CSR	20h

Function

Provides the COUT, and CFF, CFR, RRF flags.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 COUT	Analog Comparator Output Returns the current value of the analog comparator output, when read. The field is reset to 0 and reads as CCR1[COUT_INV] when the analog comparator module is disabled, that is, when CCR0[CMP_EN] = 0. Writing to this field is ignored.
7-3 —	Reserved
2 RRF	Round-Robin Flag Detects when any channel's last comparison result is different from the pre-set value in trigger mode. To clear RRF, write 1 to it. RRF is cleared when CCR0[CMP_EN] or RRCR0[RR_EN] is not set. 0b - Not detected 1b - Detected
1 CFF	Analog Comparator Flag Falling Detects when a falling edge on COUT occurs. CFF is cleared by writing 1 to it when CCR1[DMA_EN] is disabled. If CCR1[DMA_EN] is enabled, the flag is automatically cleared when DMA is done. CFF is cleared when CCR0[CMP_EN] is not set. 0b - Not detected 1b - Detected
0 CFR	Analog Comparator Flag Rising Detects when a rising edge on COUT occurs. CFR is cleared by writing 1 to it when CCR1[DMA_EN] is disabled. If CCR1[DMA_EN] is enabled, the flag is automatically cleared when DMA is done. CFR is cleared when CCR0[CMP_EN] is not set. 0b - Not detected 1b - Detected

58.7.10 Round Robin Control Register 0 (RRCR0)

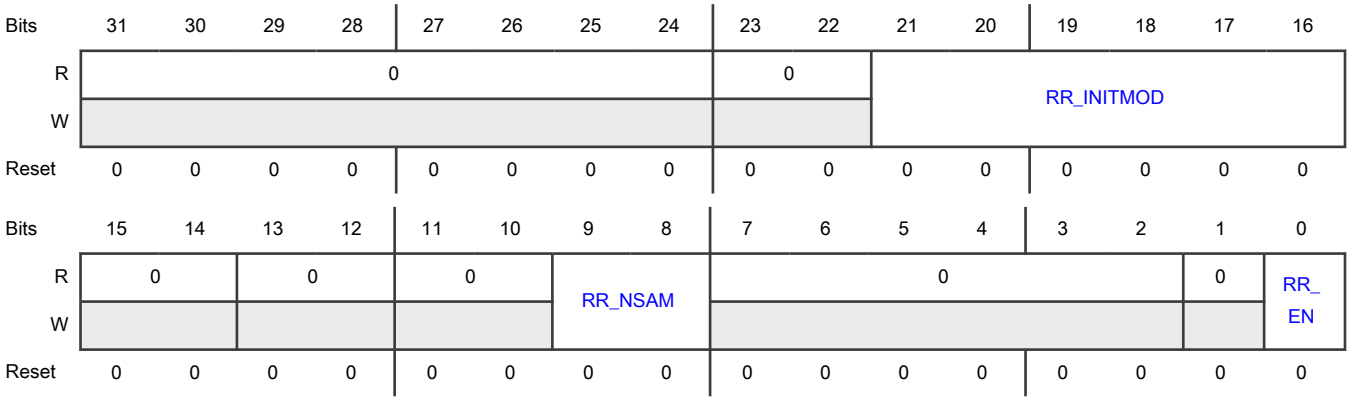
Offset

Register	Offset
RRCR0	24h

Function

Contains configuration options for round robin operation, such as enabling it and specifying the initialization delay.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-22 —	Reserved
21-16 RR_INITMOD	Initialization Delay Modulus Specifies the number of round-robin clock cycles used to determine the comparator and DAC initialization delay as specified in the chip datasheet. The initiallization delay is calculated as: RR_INITMOD * (round-robin clock period). For example, if the initialization delay is 80us and the round-robin clock is 100kHz, program RR_INITMOD to be 80us/10us = 8. 00_0000b - 63 cycles (same as 111111b) 00_0001b-11_1111b - 1 to 63 cycles
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 RR_NSAM	Number of Sample Clocks Specifies how many round-robin clock cycles to wait after the active channel is scanned before sampling the channel's comparison result. After the next cycle of the round-robin clock, the sampling takes place RR_NSAM clocks later.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - 0 clocks 01b - 1 clocks 10b - 2 clocks 11b - 3 clocks
7-2 —	Reserved
1 —	Reserved
0 RR_EN	Round-Robin Enable Enables round-robin operation. 0b - Disable 1b - Enable

58.7.11 Round Robin Control Register 1 (RRCR1)

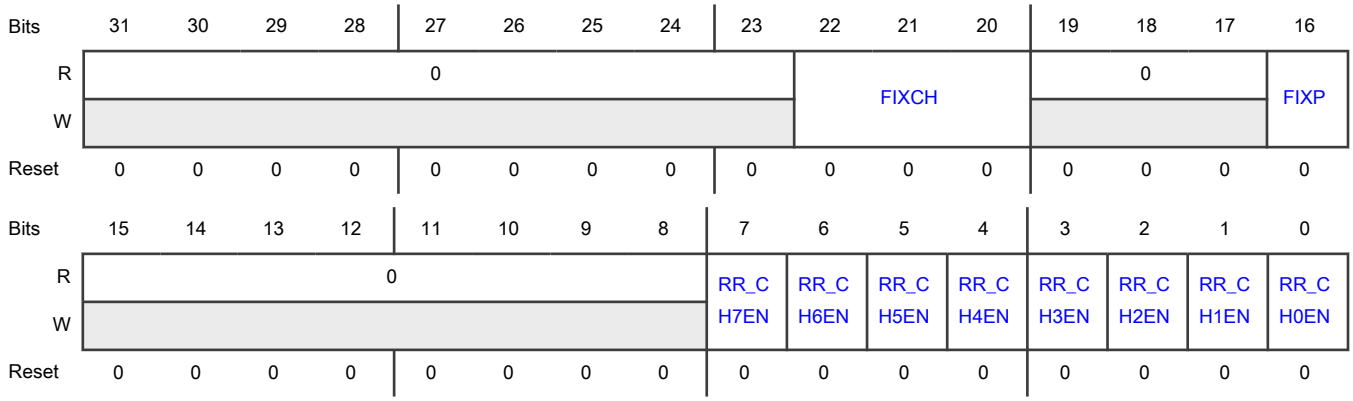
Offset

Register	Offset
RRCR1	28h

Function

Contains configuration options for round robin operation, such as enabling individual channels to participate.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-20 FIXCH	Fixed Channel Select Selects which channel in the mux port to fix for a given trigger mode application. 000b - Channel 0 001b - Channel 1 010b - Channel 2 011b - Channel 3 100b - Channel 4 101b - Channel 5 110b - Channel 6 111b - Channel 7
19-17 —	Reserved
16 FIXP	Fixed Port Fixes (locks) an analog mux port (Plus or Minus) for trigger mode. The inputs to the non-fixed port are swept during each round. 0b - Fix the Plus port. Sweep only the inputs to the Minus port. 1b - Fix the Minus port. Sweep only the inputs to the Plus port.
15-8 —	Reserved
7-0 RR_CHnEN	Channel n Input Enable in Trigger Mode Enables channel n of the non-fixed mux port to have its voltage value checked when in trigger mode. **NOTE** RR_CHnEN has no effect when the same channel is selected as the reference voltage. 0b - Disable 1b - Enable

58.7.12 Round Robin Control and Status Register (RRCSR)

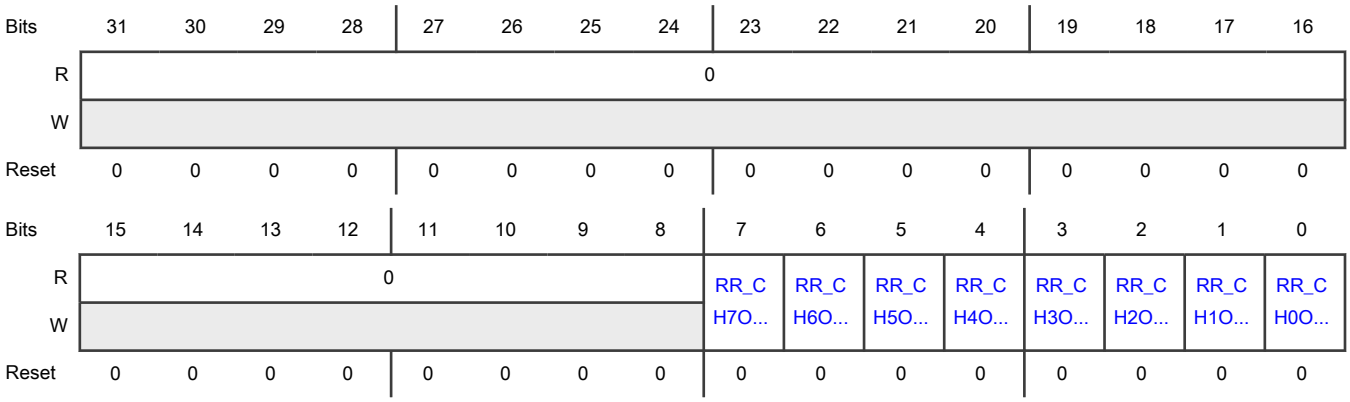
Offset

Register	Offset
RRCSR	2Ch

Function

Contains the latest comparison results of the individual channels with the fixed mux port. It also allows software to define the pre-set state for each channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 RR_CHnOUT	Comparison Result for Channel n Reading RR_CHnOUT returns the latest comparison result for channel n. Writing RR_CHnOUT defines the pre-set state for channel n.

58.7.13 Round Robin Status Register (RRSR)

Offset

Register	Offset
RRSR	30h

Function

Contains individual channel flags indicating when a channel's last comparison result is different from its pre-set value.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RR_CH7F	RR_CH6F	RR_CH5F	RR_CH4F	RR_CH3F	RR_CH2F	RR_CH1F	RR_CH0F
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 RR_CHnF	Channel n Input Changed Flag Indicates when the corresponding channel's last comparison result is different from its pre-set value. **NOTE** To clear a flag, write a 1 to it. 0b - Not different 1b - Different

58.8 Glossary

CMP	Comparator
DAC	Digital-to-analog convertor
ANMUX	Analog multiplexer

Chapter 59
Logic Control Unit (LCU)

59.1 Chip-specific LCU information

59.1.1 LCU instances

This chip has two identical LCU instances, LCU_0 and LCU_1.

59.2 Introduction

LCU selects multiple inputs from timers, pulse width modulation (PWM) signals, and input/output (I/O) pads, and combines them using a programmable logic function to create output waveforms for use by other IPs.

It consists of 3 LCs, each of which takes four inputs and creates four outputs using user-defined logic mapping. LCU also has 3 force inputs that can override the output logic and force a user-defined value directly on the output signals. Additionally, LCU has 2 sync inputs that can control when the inputs are updated to produce new output signals.

LCU generates the interrupts and DMA requests based on output signals and force input signals.

59.3 Feature list

- 3 LCs with programmable logic function for generating output results
- 12 inputs to and 12 outputs from each LC
- Muxing to map any input to any LC
- Independent filters for the rising and falling output states of each LC
- Software override logic for inputs using multiple modes
- 3 force inputs with programmable edge filtering to force output states using multiple modes
- 2 sync inputs to control transition timing
- Lock bit to block writes to configuration registers
- Interrupt request generation based on output change or force event
- DMA transfer request generation based on output change or force event

59.4 Block diagram

This figure illustrates an example implementation of LCU with:

- Three LCs
- Four force inputs
- Four sync inputs

For the number of each resource in this chip, see Feature list.

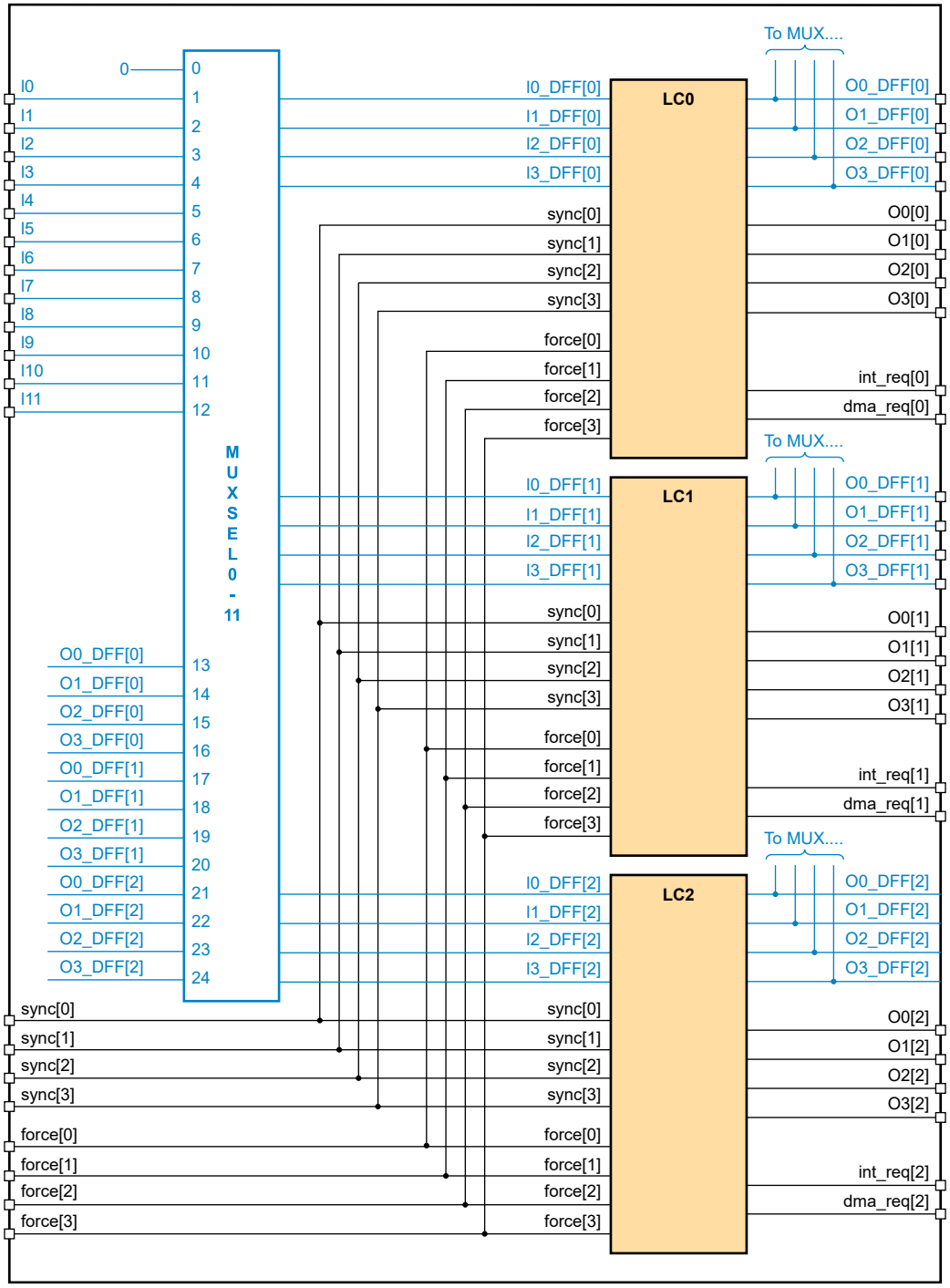


Figure 254. Block diagram

59.5 LC operation

59.5.1 Overview

An LC performs logic operations based on the LUT principle, as described in Logic operations.

Each LC supports the following features:

- Independent filter thresholds for the rising and falling output states support different transition delays.
- Bypass of the output state filter by setting the filter threshold to zero. Bypassing the filter causes the aynchronous signal to propagate to the output of the LC.
- Force logic that instantly switches an LC output to the inactive state upon external fault signal assertion. The return from the inactive state to the output state is either:
  - Instantaneous: when LCU deasserts the force signal.
  - Instantaneous synchronized: when LCU deasserts the force signal and then the sync input asserts.
  - Manual: when you write 1 to the force status bit and then the LCU deasserts the force input.
  - Manual synchronized: when you write 1 to the force status bit and LCU deasserts the force input, and then the sync input asserts.
- Input software override logic to override external inputs by writing to the following registers:
  - LC*n* Sync Control (LC*n*_SCTRL[SW_MODE])
  - Software Override Value (SWVALUE[SWVALUE])
  - Software Override Enable (SWEN[SWEN])
- Generation of interrupt requests or DMA transfer requests.

59.5.2 LC diagram

This figure illustrates an example LC implementation with:

- One LC
- Four force inputs
- Four sync inputs

For the number of each resource in this chip, see Feature list.

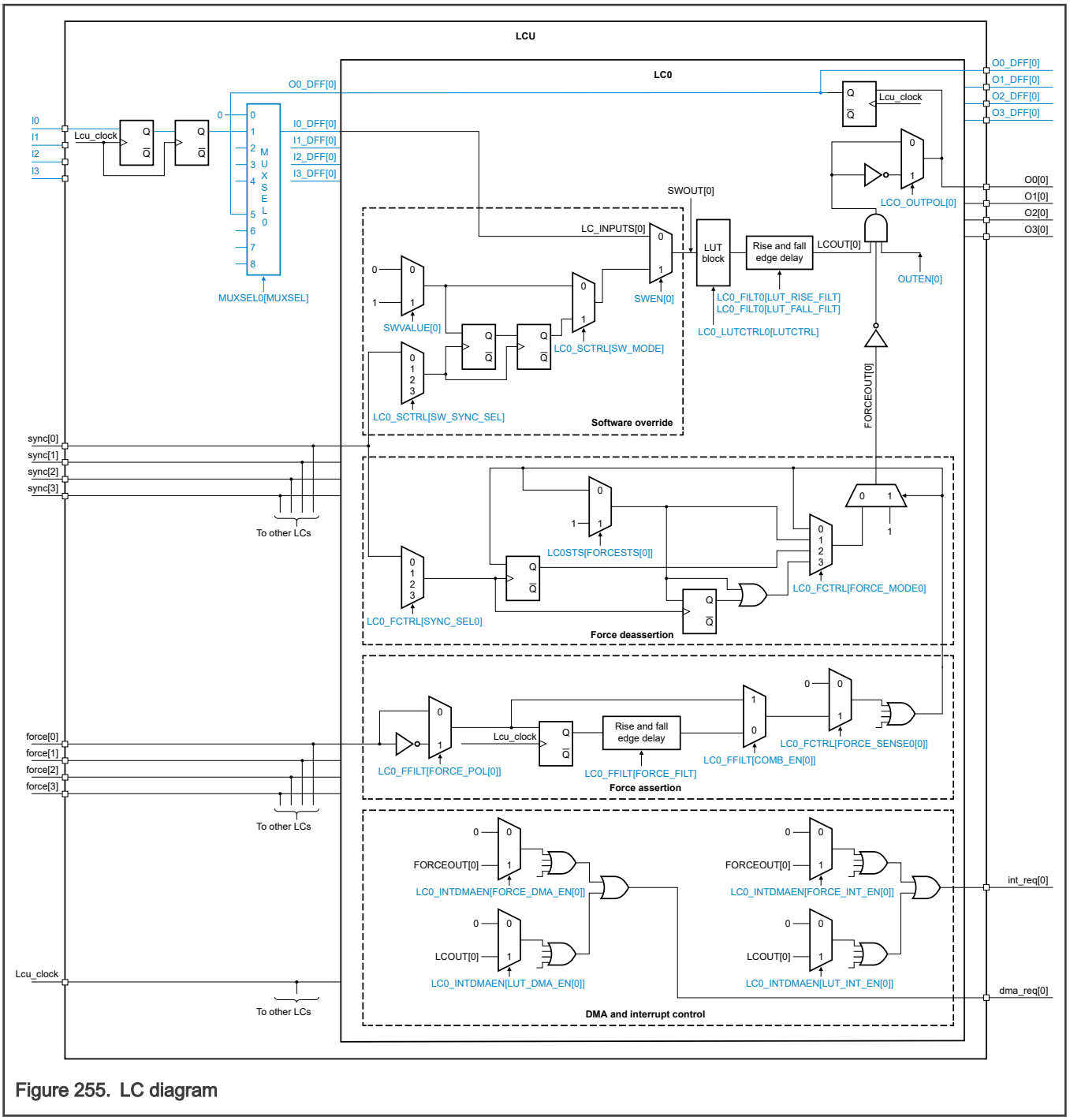


Figure 255. LC diagram

59.5.3 Logic operations

An LC reads its set of inputs (inputs 3, 2, 1, and 0) as a four-bit binary value, with Input 3 as the most-significant bit and Input 0 as the least-significant bit. For example, if:

Input 0 = 0

Input 1 = 1

Input 2 = 0

Input 3 = 1

then the combined input value is 1010b (10 decimal).

Each possible four-bit input value corresponds to a bit position in the LUT Control (LCn_LUTCTRLm[LUTCTRL]), as illustrated in LC LUT. For a given LC output, if the combined LC input value corresponds to a bit position in that output's LUTCTRL value that equals 1, then the LC asserts that output.

For example, if you write 35h to LUTCTRL (shown in the Example column of the LC LUT), then the combined input values 0 (0000b), 2 (0010b), 4 (0100b), and 5 (0101b) cause assertion of the output. All other input values cause deassertion of the output.

LCU processes each look-up table asynchronously.

59.5.4 LC LUT

The LUT maps the combined LC input value to a bit position in LC*n*_LUTCTRL*m*.

Table 284. LC LUT

Input 3	Input 2	Input 1	Input 0	LUTCTRL bit position	Example LUTCTRL value: 35h
0	0	0	0	0	1
0	0	0	1	1	0
0	0	1	0	2	1
0	0	1	1	3	0
0	1	0	0	4	1
0	1	0	1	5	1
0	1	1	0	6	0
0	1	1	1	7	0
1	0	0	0	8	0
1	0	0	1	9	0
1	0	1	0	10	0
1	0	1	1	11	0
1	1	0	0	12	0
1	1	0	1	13	0
1	1	1	0	14	0
1	1	1	1	15	0

59.5.5 LC output filters

Each LC supports two optional digital filters, a Rise Filter (LC*n*_FLIT*m*[LUT_RISE_FILT]) and a Fall Filter (LC*n*_FLIT*m*[LUT_FALL_FILT]), to post-process the LUT output and delay the output signal change.

Each filter starts accumulating clock cycles on the occurrence of the associated LUT event. The Rise Filter counts from an output assertion, and the Fall Filter counts from an output deassertion. When the number of clock cycles reaches the filter threshold (matches the specified filter value), the output flips to the new state. For example, if you specify a Rise Filter value of 100h, and the LUT event triggers an output assertion, the output signal does not assert until the 256th clock cycle after the event.

A filter value of 0 bypasses the filter.

59.6 Behavior in different chip modes of operation

Operation mode	Status of outputs
Normal	• Controlled by LC*n*_LUTCTRL*m* and MUXSEL*n* • Toggled based on the states of the selected inputs
Debug	Inactive or normal operation controlled by Debug Mode Enable (DBGEN)

59.7 Use-case examples

59.7.1 Two-channel multiplexer

59.7.1.1 Overview

A multiplexer (or mux) takes multiple inputs and, based on some control mechanism, passes only one of those inputs to the output. You can implement a two-channel multiplexer controlled by an external signal or a software override using three inputs and one output of a single LC, as illustrated in Logic diagram and Implementation.

This example implementation controls the multiplexer with the SEL input signal and a software override. The software override controls the multiplexer output regardless of the SEL signal state. To use software override, you must enable it for the corresponding output (SWEN[SWEN]). You assert software override by writing 1 to the corresponding bit of the Software Override Value (SWVALUE[SWVALUE]).

59.7.1.2 Logic diagram

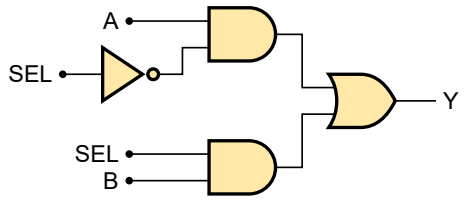


Figure 256. Logic diagram

59.7.1.3 Implementation

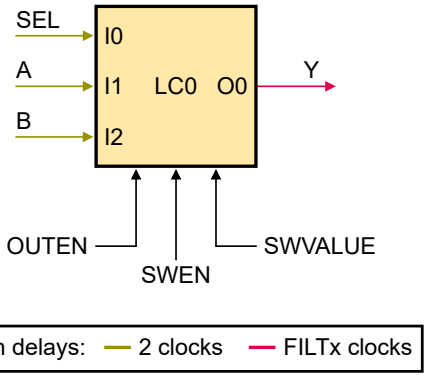


Figure 257. Implementation

59.7.1.4 Connections and controls

Table 285. Connections and controls

Name	Description
SEL	Input signal
A	
B	
OUTEN	Output enable
SWEN	Software override enable
SWVALUE	Software override
Y	Output signal

59.7.1.5 Truth table

Table 286. Truth table

Inputs			Output
A	B	SEL	Y
0	0	0	0
0	0	1	0
1	0	0	1
1	0	1	0
0	1	0	0
0	1	1	1
1	1	0	1
1	1	1	1

59.7.1.6 Register configuration

Table 287. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	E4E4h	—	—	—
LUTCTRL1	—	—	—	—	—	0h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h

Table continues on the next page...

Table 287. Register configuration (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	3h	—	—	—	—	—	—
MUXSEL3	0h	—	—	—	—	—	—	—
SWEN	—	—	—	—	Disable software override for LC0 input 0: 0h Enable software override for LC0 input 0: 1h			
SWVALUE	—	—	—	—	Deassert software override for LC0 input 0: 0h Assert software override for LC0 input 0: 1h			
OUTEN	—	—	—	—	1h			

59.7.1.7 Waveforms

In this figure, the SWEN and SWVALUE signals represent the states of the software override control registers. Writes to SWEN and SWVALUE have immediate impact on LC outputs, whereas LC inputs require input synchronization to prevent metastability.

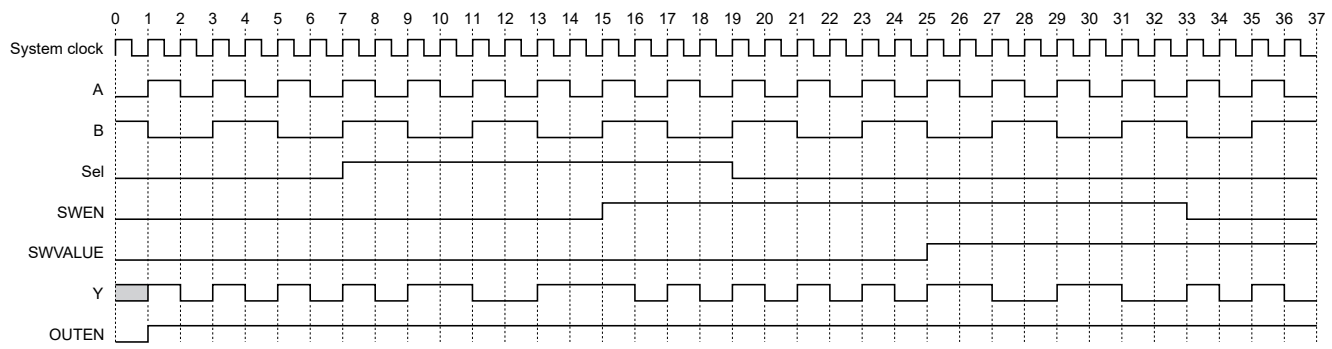


Figure 258. Waveforms

59.7.2 Binary to Gray code converter

59.7.2.1 Overview

Gray code, also known as reflected binary code (RBC), is a non-weighted, minimum change code. In this code, any two adjacent code numbers differ by only one bit. This code solves the problem of physical switch transitions by changing only one switch at a time, so there is never any ambiguity of position.

You can implement binary to Gray code conversion using a single LC, as illustrated in Logic diagram and Implementation.

In this example, the LC also generates a DMA request on each rising edge of output 3, which is bit 0 of the Gray code (G0).

59.7.2.2 Conversion table

Table 288. Conversion table

Decimal	Binary	Gray
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

59.7.2.3 Logic diagram

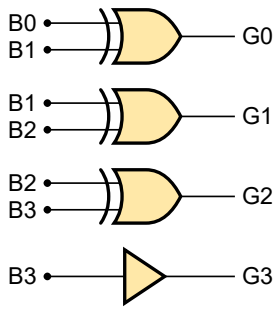


Figure 259. Logic diagram

59.7.2.4 Implementation

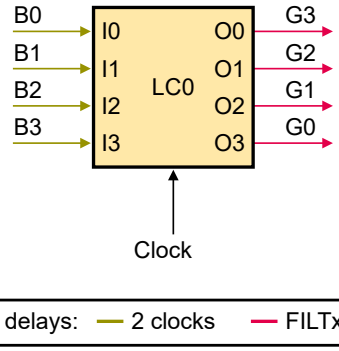


Figure 260. Implementation

59.7.2.5 Connections and controls

Table 289. Connections and controls

Name	Description
B0	Bit 0 of the input binary code
B1	Bit 1 of the input binary code
B2	Bit 2 of the input binary code
B3	Bit 3 of the input binary code
G0	Bit 0 of the output Gray code
G1	Bit 1 of the output Gray code
G2	Bit 2 of the output Gray code
G3	Bit 3 of the output Gray code
Clock	System clock input

59.7.2.6 Truth table

Table 290. Truth table

Inputs				Outputs			
B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1

Table continues on the next page...

Table 290. Truth table (continued)

Inputs				Outputs			
B3	B2	B1	B0	G3	G2	G1	G0
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

59.7.2.7 Register configuration

Table 291. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	FF00h	—	—	—
LUTCTRL1	—	—	—	—	—	FF0h	—	—
LUTCTRL2	—	—	—	—	—	—	3C3Ch	—
LUTCTRL3	—	—	—	—	—	—	—	6666h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	800h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0h	—	—	—	01h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	3h	—	—	—	—	—	—
MUXSEL3	4h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			

Table continues on the next page...

Table 291. Register configuration (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
OUTEN	—	—	—	—	Fh			

59.7.2.8 Waveforms

The LC outputs are delayed by two clock cycles because of the two-stage input synchronizers.

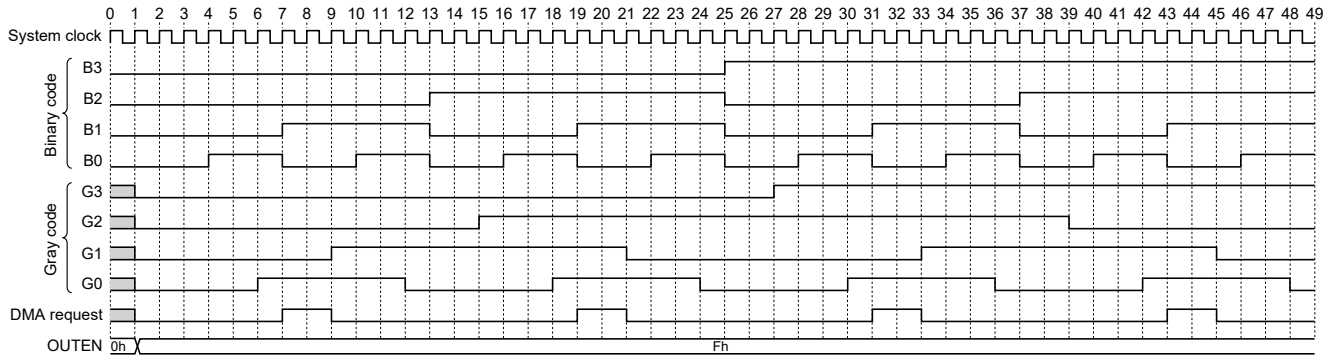


Figure 261. Waveforms

59.7.3 Manchester encoder and decoder

59.7.3.1 Overview

Manchester encoding (also known as phase encoding) is a data-modulation technique for binary data transfer based one of these signals:

- Analog
- RF
- Optical
- High-speed-digital
- Long-distance-digital

The fundamental idea behind Manchester encoding is to use voltage transitions, instead of voltage levels, to represent 1s and 0s. It performs an exclusive OR of data and clock signals to encode and decode data. A high-to-low transition on the falling clock edge indicates 0. A low-to-high transition on the rising clock edge indicates 1.

59.7.3.2 Logic diagram

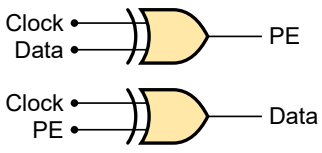


Figure 262. Logic diagram

59.7.3.3 Implementation

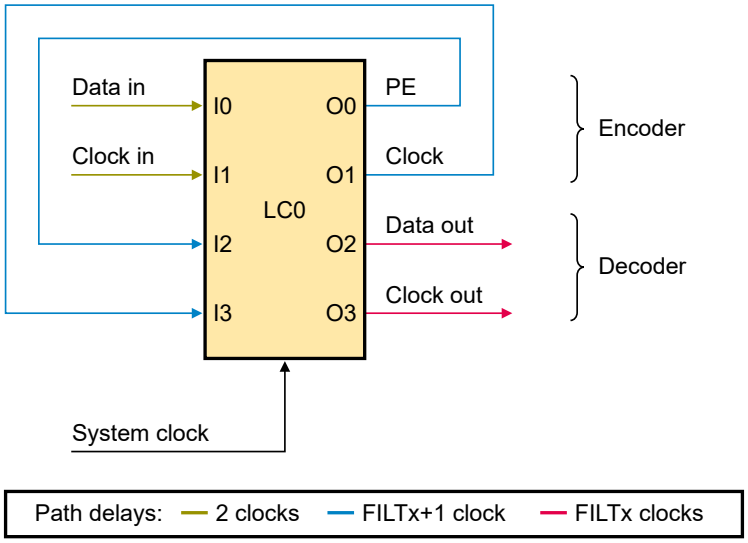


Figure 263. Implementation

59.7.3.4 Connections and controls

Table 292. Connections and controls

Name	Description
Data in	Input signal
Clock in	
PE	Manchester-encoded signal from LC
Clock	Auxiliary signal fed to the decoder to allow Manchester decoding
Data out	Output signal after Manchester decoding
System clock	Clock frequency supplied to LC
Clock out	Output signal

59.7.3.5 Truth table for encoder

Table 293. Truth table for encoder

Inputs		Output
Data in	Clock in	PE
0	0	0
0	1	1
1	0	1
1	1	0

59.7.3.6 Truth table for decoder

Table 294. Truth table for decoder

Inputs		Output
Clock	PE	Data out
0	0	0
0	1	1
1	0	1
1	1	0

59.7.3.7 Register configuration

Table 295. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	6666h	—	—	—
LUTCTRL1	—	—	—	—	—	AAAAh	—	—
LUTCTRL2	—	—	—	—	—	—	FF0h	—
LUTCTRL3	—	—	—	—	—	—	—	FF00h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	5h	—	—	—	—	—	—
MUXSEL3	6h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	Fh			

59.7.3.8 Waveforms for encoder

This figure includes the Manchester-encoded waveform (labeled PE) of the input waveform (labeled "Data in"). The PE waveform is delayed by two clock cycles because of two-stage input synchronization.

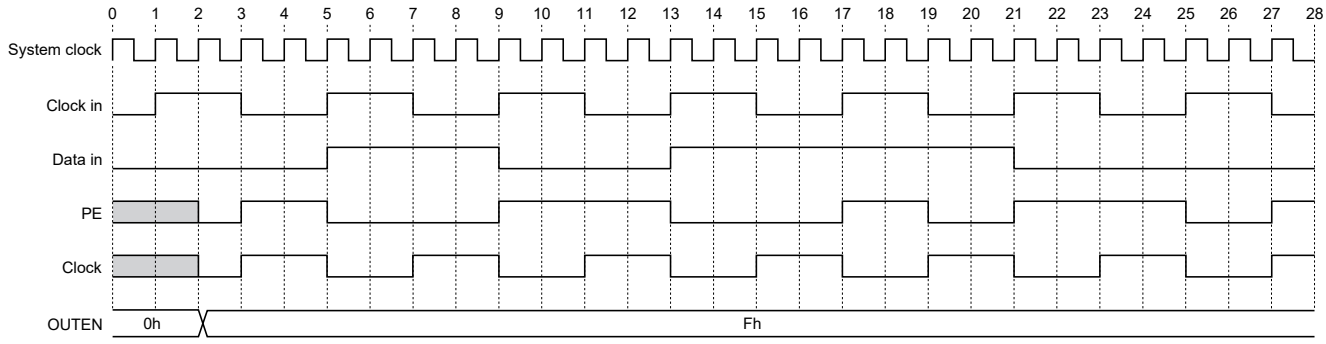


Figure 264. Waveforms for encoder

59.7.3.9 Waveforms for decoder

The output waveform (labeled "Data out") is delayed by four clock cycles from the input waveform (labeled "Data in") because of two-stage input synchronization. In this example, the Manchester encoder also propagates the Clock waveform that the decoder uses to reconstruct the "Data in" waveform from the Manchester-decoded waveform (labeled PE).

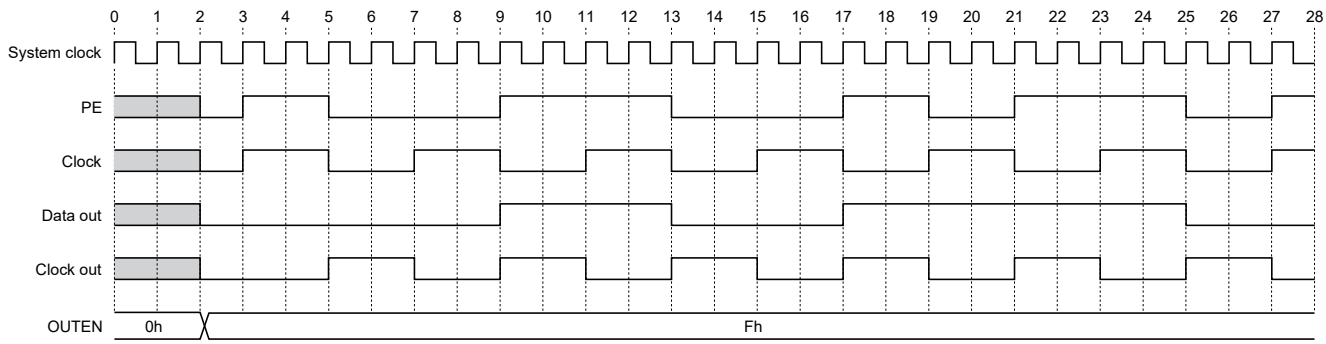


Figure 265. Waveforms for decoder

59.7.4 Adjustable PWM/FM generator

59.7.4.1 Overview

In many cases, it is useful to route an LC output back to one of its inputs, as illustrated in Implementation. In addition, you can delay outputs with a digital filter with different time constants for delaying rising and falling edges. Combining these capabilities, you can use an LC to generate output waveforms with adjustable pulse width and frequency.

In addition to waveform generation, the software-override feature forces the generator output low or high by software. You control the override via Software Override Enable (SWEN) and Software Override Value (SWVALUE).

59.7.4.2 Implementation

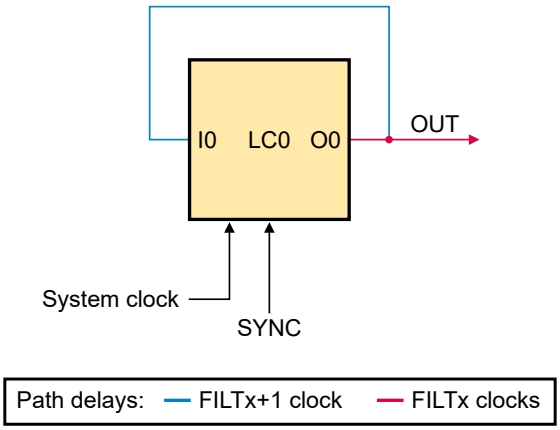


Figure 266. Implementation

59.7.4.3 Connections and controls

Table 296. Connections and controls

Name	Description
I0	Delayed output routed back to LC input
O0 and OUT	Delayed output
System clock	Clock frequency supplied to LC
SYNC	Software override synchronization signal

59.7.4.4 Truth table

This example generates an adjustable PWM/FM waveform according to the following table. In the table:

- OUT is the LC output signal that is also routed to the LC input using an internal multiplexer.
- TdH is the digital filter value to delay the rising edge of the output signal.
- TdL is the digital filter value to delay the falling edge of the output signal.

Table 297. Truth table

I0	OUT
0	1 delayed by TdH (see Output edge delays)
1	0 delayed by TdL (see Output edge delays)

59.7.4.5 Output edge delays

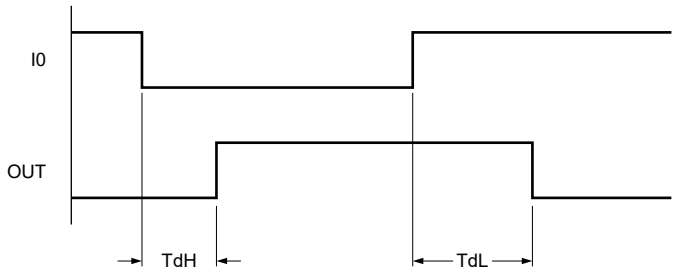


Figure 267. Output edge delays

59.7.4.6 Register configuration

Table 298. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	1h	—	—	—
LUTCTRL1	—	—	—	—	—	0h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	1_0001h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	SWVALUE for input 0 changes immediately: 0h SWVALUE for input 0 changes on rising edge of sync: 1h			
MUXSEL0	—	—	—	5h	—	—	—	—
MUXSEL1	—	—	0h	—	—	—	—	—
MUXSEL2	—	0h	—	—	—	—	—	—
MUXSEL3	0h	—	—	—	—	—	—	—
SWEN	—	—	—	—	Disable software override: 0h Enable software override: 1h			
SWVALUE	—	—	—	—	Deassert software override: 0h			

Table continues on the next page...

Table 298. Register configuration (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
					Assert software override: 1h			
OUTEN	—	—	—	—	1h			

59.7.4.7 Waveforms for immediate software override

In this figure, the SWEN and SWVALUE signals represent the states of the software override control registers. Writes to Software Override Enable (SWEN) and Software Override Value (SWVALUE) have immediate impact on LC outputs, whereas LC inputs require input synchronization to prevent metastability.

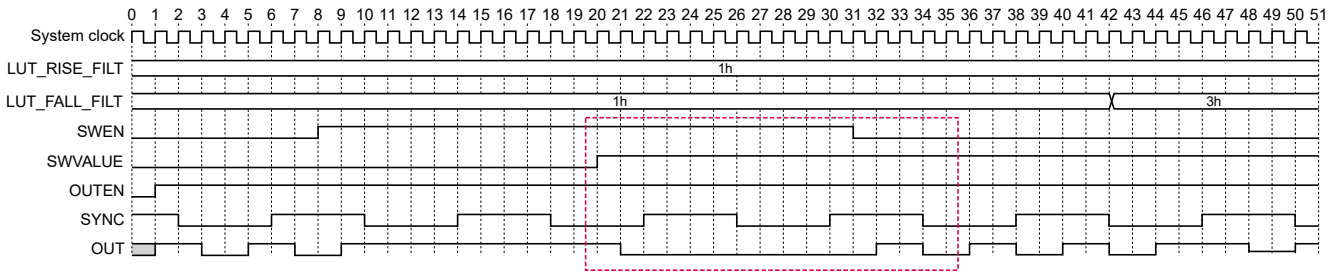


Figure 268. Waveforms for immediate software override

59.7.4.8 Waveforms for synced software override

In this example, software override control is synced with the rising edges of the SYNC waveform.

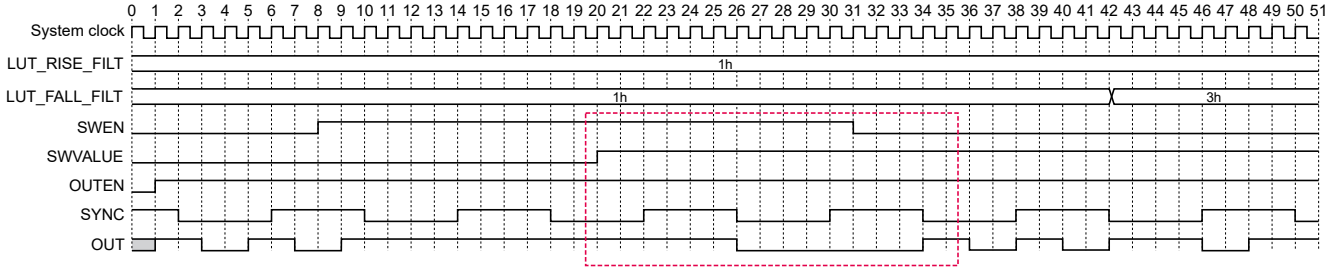


Figure 269. Waveforms for synced software override

59.7.5 SR latch

59.7.5.1 Overview

An SR latch is the simplest bistable device and a fundamental component of data storage.

This example implementation uses one LC that includes an internal feedback path configured by the internal multiplexer, as illustrated in Implementation. Output Q responds to inputs S (set) and R (reset). It changes with a delay of two clock cycles—the time required for the S and R signal inputs to pass through the internal two-stage synchronizers. This example implementation bypasses the Q digital filter; therefore, the loop of Q back to the input experiences a delay of one clock cycle." See Waveforms.

59.7.5.2 Implementation

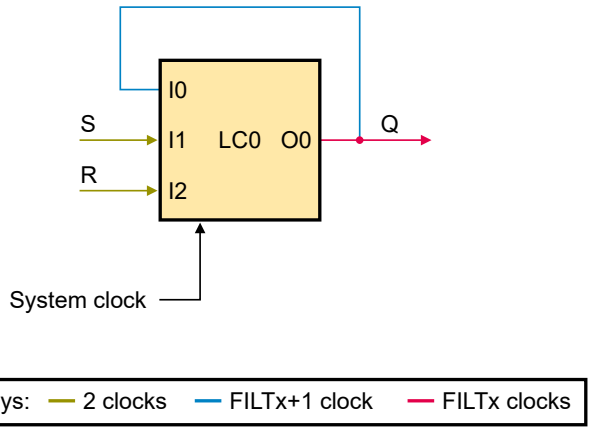


Figure 270. Implementation

59.7.5.3 Connections and controls

Table 299. Connections and controls

Name	Description
S	Set input
R	Reset input
Q	Both an output signal and an input that feeds back to LC0
System clock	Clock frequency supplied to LC

59.7.5.4 Truth table

Table 300. Truth table

Inputs			Output	State
R	S	Q	Q	
0	0	0	0	Latch
0	0	1	1	Latch
0	1	0	1	Set
0	1	1	1	Set
1	0	0	0	Reset
1	0	1	0	Reset
1	1	0	0	Not allowed
1	1	1	0	

59.7.5.5 Register configuration

Table 301. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	1h	—	—	—
LUTCTRL1	—	—	—	—	—	0h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	5h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	3h	—	—	—	—	—	—
MUXSEL3	0h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	1h			

59.7.5.6 Waveforms

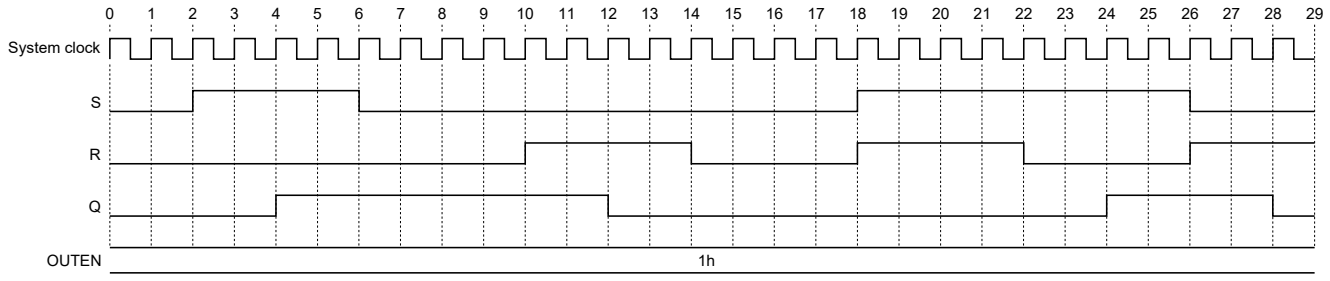


Figure 271. Waveforms

59.7.6 D flip-flop

59.7.6.1 Overview

A D flip-flop stores and outputs whatever logic level is applied to its data terminal at the time its clock input goes high.

You can implement a D flip-flop element using an LC as illustrated in Implementation. On a clock rising edge, the LC either asserts or deasserts the Q output based on the D input.

The Q output and CLKO digital filters are bypassed. Therefore, the states of these variables are fed to inputs with a delay of one clock cycle. See Waveforms.

59.7.6.2 Implementation

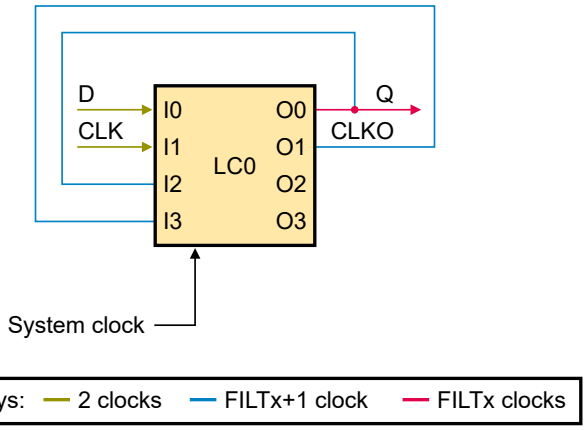


Figure 272. Implementation

59.7.6.3 Connections and controls

Table 302. Connections and controls

Name	Description
D	Input signal
CLK	Input clock
Q	Output signal; holds the logic state of the last Q output
CLKO	Holds the logic state of the last CLK input

59.7.6.4 Truth table

Table 303. Truth table

Inputs				Outputs		State
CLKO	Q	CLK	D	Q	CLKO	
0	0	0	0	0	0	Hold
0	0	0	1	0	0	Hold
0	0	1	0	0	1	Reset
0	0	1	1	1	1	Set
0	1	0	0	1	0	Hold

Table continues on the next page...

Table 303. Truth table (continued)

Inputs				Outputs		State
CLKO	Q	CLK	D	Q	CLKO	
0	1	0	1	1	0	Hold
0	1	1	0	0	1	Reset
0	1	1	1	1	1	Set
1	0	0	0	0	0	Hold
1	0	0	1	0	0	Hold
1	0	1	0	0	1	Hold
1	0	1	1	0	1	Hold
1	1	0	0	1	0	Hold
1	1	0	1	1	0	Hold
1	1	1	0	1	1	Hold
1	1	1	1	1	1	Hold

59.7.6.5 Register configuration

Table 304. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	F0B8h	—	—	—
LUTCTRL1	—	—	—	—	—	CCCCh	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	5h	—	—	—	—	—	—
MUXSEL3	6h	—	—	—	—	—	—	—

Table continues on the next page...

Table 304. Register configuration (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	3h			

59.7.6.6 Waveforms

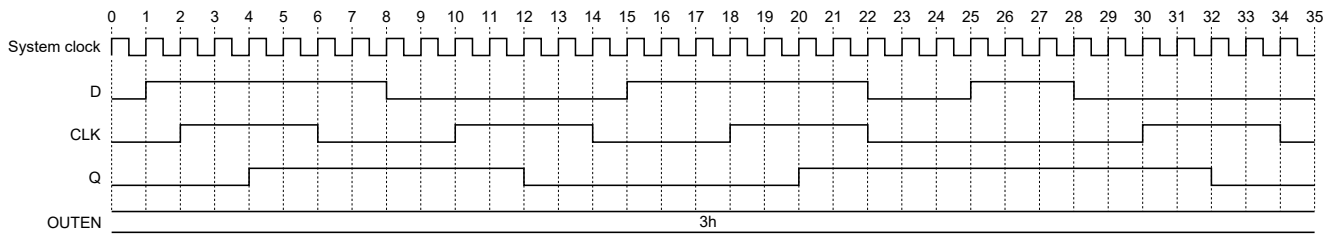


Figure 273. Waveforms

59.7.7 JK flip-flop

59.7.7.1 Overview

A JK flip-flop is a gated SR latch (see SR latch) with an added clock input circuit that prevents the illegal or invalid output condition that can occur when both SR-latch inputs (S and R) are asserted.

Truth table for JK flip-flop presents the four possible input combinations for a JK flip-flop. All output changes occur at the clock rising edge.

You can create a JK flip-flop using two LCs, as illustrated in Implementation. The first LC detects rising edges of the CLK clock input (see Truth table for LC0). The second LC processes J and K inputs at every rising clock edge (see Truth table for LC1).

Output Q and CLKO digital filters are bypassed. Therefore, the states of these variables are fed to inputs with a delay of one clock cycle (see Waveforms).

59.7.7.2 Implementation

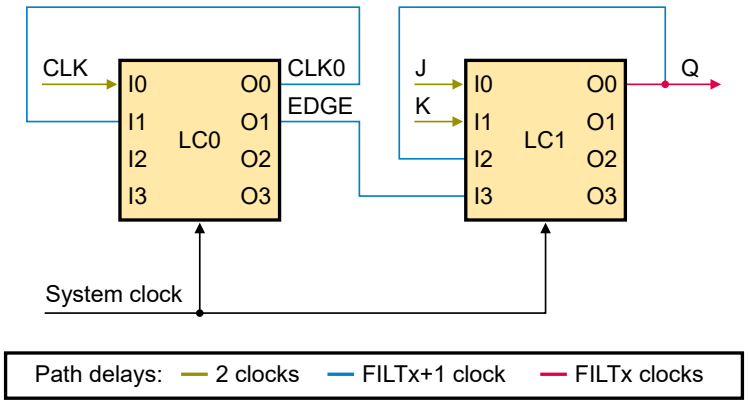


Figure 274. Implementation

59.7.7.3 Truth table for JK flip-flop

Table 305. Truth table for JK flip-flop

J	K	State
0	0	Hold
1	0	Set
0	1	Clear
1	1	Toggle

59.7.7.4 Truth table for LC0

Table 306. Truth table for LC0

Inputs		Outputs		State
CLKO	CLK	CLKO	EDGE	
0	0	0	0	No edge
0	1	1	1	Edge detected
1	0	0	0	No edge
1	1	1	0	No edge

59.7.7.5 Truth table for LC1

Table 307. Truth table for LC1

Inputs				Output	State
EDGE	Q	J	K	Q	
0	0	X	X	0	Hold
0	1	X	X	1	Hold
1	0	0	0	0	Hold
1	1	0	0	1	Hold
1	X	1	0	1	Set
1	X	0	1	0	Clear
1	0	1	1	1	Toggle
1	1	1	1	0	Toggle

59.7.7.6 Register configuration for LC0

Table 308. Register configuration for LC0

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	Ah	—	—	—
LUTCTRL1	—	—	—	—	—	2h	—	—

Table continues on the next page...

Table 308. Register configuration for LC0 (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	9h	—	—	—	—	—
MUXSEL2	—	0h	—	—	—	—	—	—
MUXSEL3	0h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	3h			

59.7.7.7 Register configuration for LC1

Table 309. Register configuration for LC1

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	3AF0h	—	—	—
LUTCTRL1	—	—	—	—	—	0h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			

Table continues on the next page...

Table 309. Register configuration for LC1 (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL4	—	—	—	5h	—	—	—	—
MUXSEL5	—	—	6h	—	—	—	—	—
MUXSEL6	—	Dh	—	—	—	—	—	—
MUXSEL7	Ah	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	10h			

59.7.7.8 Waveforms

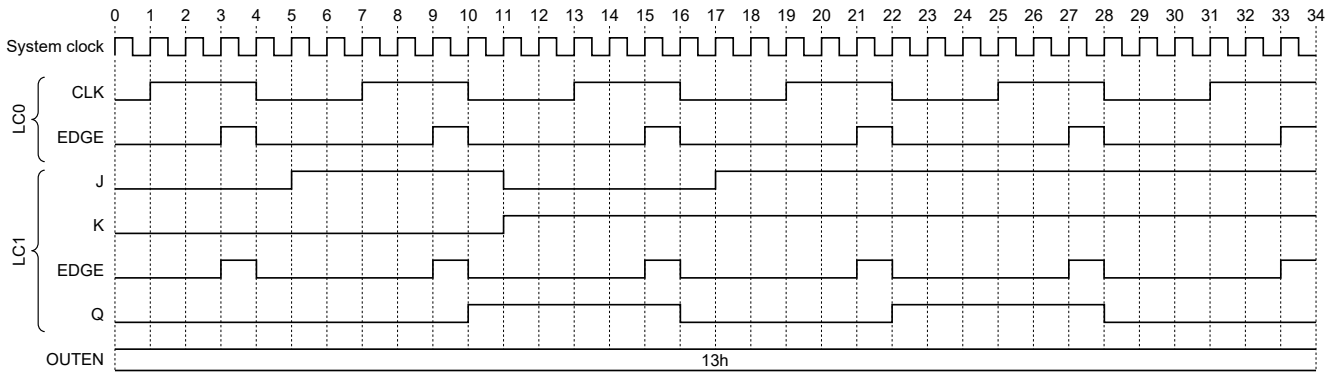


Figure 275. Waveforms

59.7.8 Incremental encoder

59.7.8.1 Overview

An incremental encoder provides A and B pulse outputs in the form of 90-degree shifted waveforms (see Outputs). The decoding circuit processes these waveforms to detect the rotor position and the direction of rotation.

This example implementation requires two LCs (see Implementation):

- LC0 provides pulse streams on CW or CCW output at every A and B signal edge. If the encoder rotation is CW (A signal leading), then pulses appear at the CW output. If the encoder rotation is CCW (B signal leading), then pulses appear at the CCW output. Two pulse counters accumulate these pulse streams to provide the actual absolute encoder position in the respective direction of rotation.
- LC1 then uses the LC0 outputs to detect the direction of the rotation.

59.7.8.2 Outputs

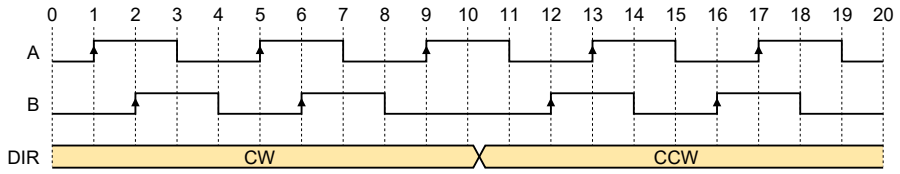


Figure 276. Outputs

59.7.8.3 Implementation

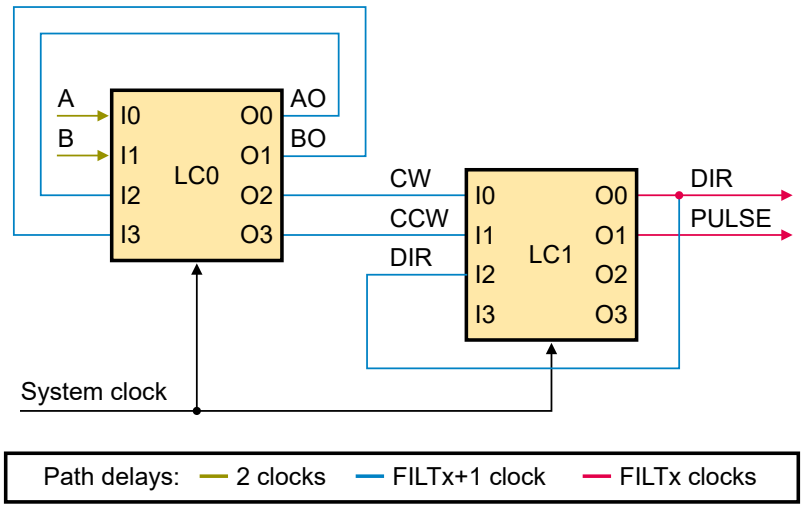


Figure 277. Implementation

59.7.8.4 Truth table for pulse generation

Table 310. Truth table for pulse generation

Inputs				Outputs			
BO	AO	B	A	AO	BO	CW	CCW
0	0	0	0	0	0	—	—
0	0	0	1	1	0	1	—
0	0	1	0	0	1	—	1
0	0	1	1	1	1		
0	1	0	0	0	0	—	1
0	1	0	1	1	0		
0	1	1	0	0	1	—	—
0	1	1	1	1	1	1	—
1	0	0	0	0	0	1	

Table continues on the next page...

Table 310. Truth table for pulse generation (continued)

Inputs				Outputs			
BO	AO	B	A	AO	BO	CW	CCW
1	0	0	1	1	0	—	—
1	0	1	0	0	1	—	—
1	0	1	1	1	1	—	1
1	1	0	0	0	0		
1	1	0	1	1	0	—	1
1	1	1	0	0	1	1	—
1	1	1	1	1	1	—	—

59.7.8.5 Truth table for decoding direction of rotation

In this table:

- DIR = 0 represents a CCW direction.
- DIR = 1 represents a CW direction.

Table 311. Truth table for decoding direction of rotation

Inputs			Output
DIR	CCW	CW	DIR
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

59.7.8.6 Register configuration for LC0

Table 312. Register configuration for LC0

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	AAAAh	—	—	—
LUTCTRL1	—	—	—	—	—	CCCCh	—	—
LUTCTRL2	—	—	—	—	—	—	4182h	—
LUTCTRL3	—	—	—	—	—	—	—	2814h
FILT0	—	—	—	—	1_0001h	—	—	—

Table continues on the next page...

Table 312. Register configuration for LC0 (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
FILT1	—	—	—	—	—	1_0001h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	9h	—	—	—	—	—	—
MUXSEL3	Ah	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	Fh			

59.7.8.7 Register configuration for LC1

Table 313. Register configuration for LC1

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	BAh	—	—	—
LUTCTRL1	—	—	—	—	—	0h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL4	—	—	—	Bh	—	—	—	—

Table continues on the next page...

Table 313. Register configuration for LC1 (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
MUXSEL5	—	—	Ch	—	—	—	—	—
MUXSEL6	—	Dh	—	—	—	—	—	—
MUXSEL7	0h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	10h			

59.7.8.8 Waveforms

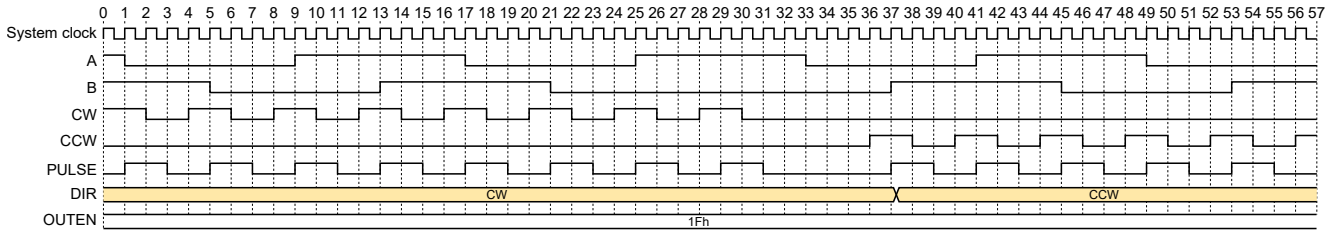


Figure 278. Waveforms

59.7.9 AC motor PWM controller

59.7.9.1 Overview

Driving a three-phase AC motor requires dedicated PWMs to perform:

- Complementary signal generation
- Dead time insertion
- Fault-state assertion
- Manual or automatic fault recovery

LCU enhances basic timer functionality with the necessary complementary features available only on dedicated motor control PWMs.

Implementation for AC motor controller shows all input and output signals of LCU in a typical three-phase motor control application.

TdH is the time delay preceding each rising edge of the complementary PWM output signals. See Truth table for details.

Fault signals are routed from analog comparators or external protection circuits to protect power switches against damage caused by overcurrent. Any active fault must immediately change all PWM outputs to the inactive state. Force logic causes PWM outputs to transition to the inactive state upon an external fault event. See Truth table for details.

The dead time is one clock cycle. On a fault event, both PWM outputs transition to the inactive state (logic 0). When the fault event deasserts, both PWM signals start operating according to FORCE_MODE = 00 (cleared on force deassertion). See Waveforms for force cleared on force deassertion for details.

59.7.9.2 Implementation for AC motor controller

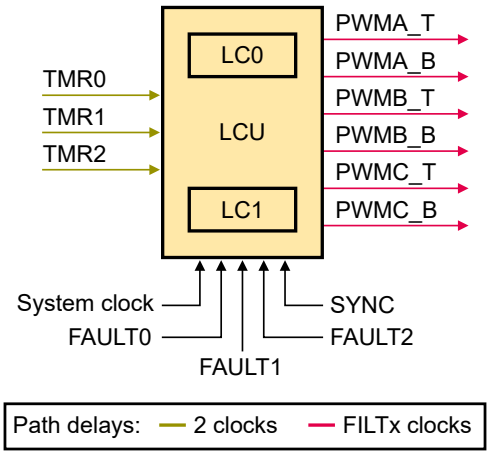


Figure 279. Implementation for AC motor controller

59.7.9.3 Implementation for controlling one phase of AC motor controller

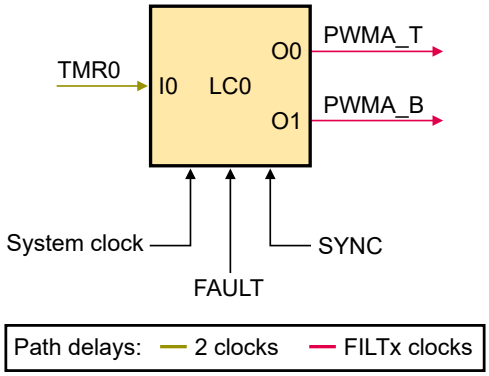


Figure 280. Implementation for controlling one phase of AC motor controller

59.7.9.4 Connections and controls

Table 314. Connections and controls

Name	Description
TMR*n*	Synchronized timer input signals
System clock	Clock supplied to the LC
FAULT*n*	Fault signal (see Overview)
SYNC	Sync signal for optional automatic fault recovery
PWM*x*_T	PWM output signal for controlling top power switch
PWM*x*_B	PWM output signal for controlling bottom power switch

59.7.9.5 Truth table

Table 315. Truth table

Input	Outputs	
TMR0	PWMA_T	PWMB_B
0	0	1 delayed by TdH (see Output edge delays)
1	1 delayed by TdH (see Output edge delays)	0

59.7.9.6 Register configuration

Table 316. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	2h	—	—	—
LUTCTRL1	—	—	—	—	—	1h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	1_0000h	—	—	—
FILT1	—	—	—	—	—	1_0000h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	100_0000h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	Software sync on rising edge for inputs 2 and 0, immediate for 3 and 1: 101h Software sync on rising edge for all four inputs: 1111h			
MUXSEL0	—	—	—	01h	—	—	—	—
MUXSEL1	—	—	0h	—	—	—	—	—
MUXSEL2	—	0h	—	—	—	—	—	—
MUXSEL3	0h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	3h			

59.7.9.7 Waveforms for force cleared on force deassertion

This example shows two behaviors, depending on the Force Clearing mode (LC0_FCTRL[FORCE_MODE0]).

00b - Both PWM signals start normal operation when the fault event deasserts. See Waveforms for force cleared on force deassertion for details.

01b - Both PWM signals start normal operation on rising sync after the fault event deasserts. See Waveforms for force cleared on rising sync after force deassertion for details.

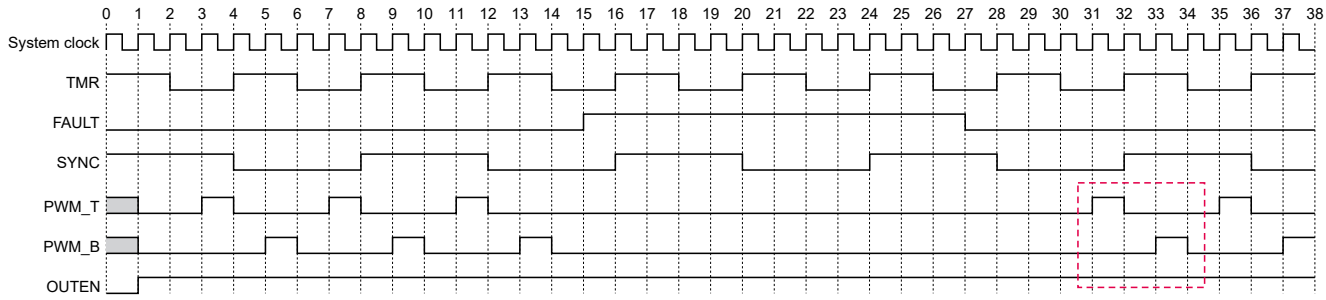


Figure 281. Waveforms for force cleared on force deassertion

59.7.9.8 Waveforms for force cleared on rising sync after force deassertion

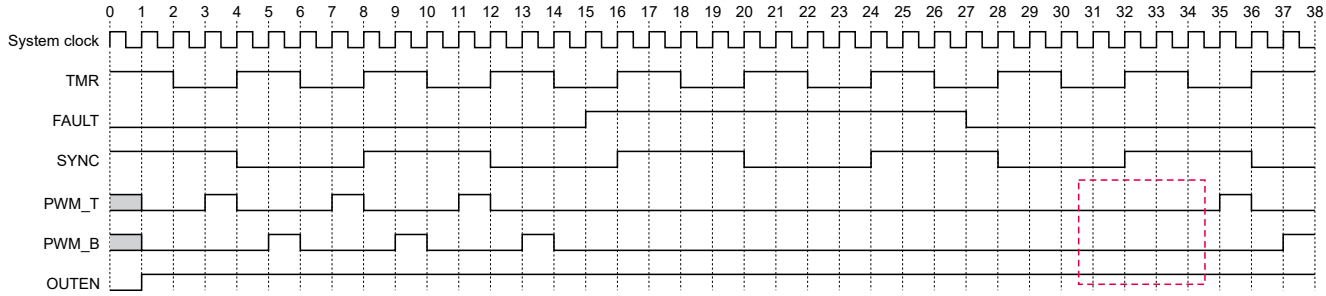


Figure 282. Waveforms for force cleared on rising sync after force deassertion

59.7.10 AC motor double-switching controller

59.7.10.1 Overview

The simplest method of obtaining motor winding currents is to measure the current in each phase by placing a shunt resistor in two or three inverter legs (phases).

This example implementation of an AC motor drive for low-cost and high-volume applications uses a single shunt on the DC bus return path, as illustrated in AC motor power stage schematic. LCU supports double-switching by XORing two timer channel outputs. The implementation uses two LCs arranged to perform double switching, dead time generation, and the processing of three external fault signals.

59.7.10.2 Double-switching

The double-switching technique requires:

- Six synchronized timer pulses
- Three fault signals
- Six PWM outputs for controlling top and bottom power switches
- A sync signal for automatic fault recovery

Waveforms for double-switching technique example applies the PWM switching cycle to an AC motor to reconstruct currents in three phases using a single DC bus shunt resistor (RSH). It applies the non-zero switching vectors 100, 110, and 101 to the power switches, which sample currents IA[0], -IC[0], -IC[1], and IA[1]. Because the durations (T1) of active vectors 110 and 101 are the same, it also applies the zero switching vector (000) with durations T2 and T3 in the middle of the switching cycle to

allow phase current reconstruction. This technique can only be used by PWM modules targeted at motor control and power conversion applications.

59.7.10.3 Fault inputs and deadtime insertion

Fault signals are routed from analog comparators or external protection circuits to protect power switches against damage caused by overcurrent. You must program the Combinational Force Path (LCn_FFILT[COMB_EN]) to enable an active fault input to immediately force all PWM outputs to the inactive state.

59.7.10.4 Deadtime insertion

In Waveforms for generation of PWM complementary signals using an LC, the PWMA_T output is generated by XORing timer outputs TMR0 and TMR1. The PWMA_B output is complementary to PWMA_T. Digital filters insert a dead time of one system clock cycle to delay the rising edges of both PWM signals.

59.7.10.5 Fault events

Waveforms for force cleared on force deassertion shows the reaction to the fault event. On a fault event, both the PWM outputs transition to the inactive state (logic 0). When fault event deasserts, both the PWM signals start operating according to FORCE_MODE = 00 (cleared on force deassertion).

Alternatively, after the fault event deasserts, fault clearing can be performed according to FORCE_MODE = 01 (cleared on rising sync force deassertion). See Waveforms for force cleared on rising sync after force deassertion for details.

59.7.10.6 AC motor power stage schematic

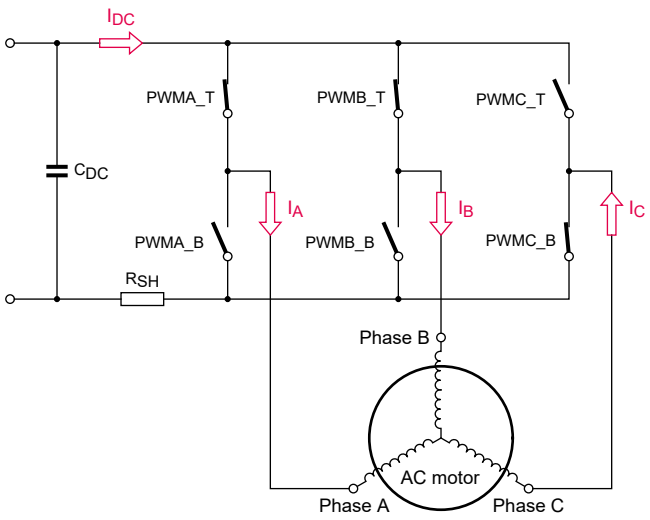


Figure 283. AC motor power stage schematic

59.7.10.7 Waveforms for double-switching technique example

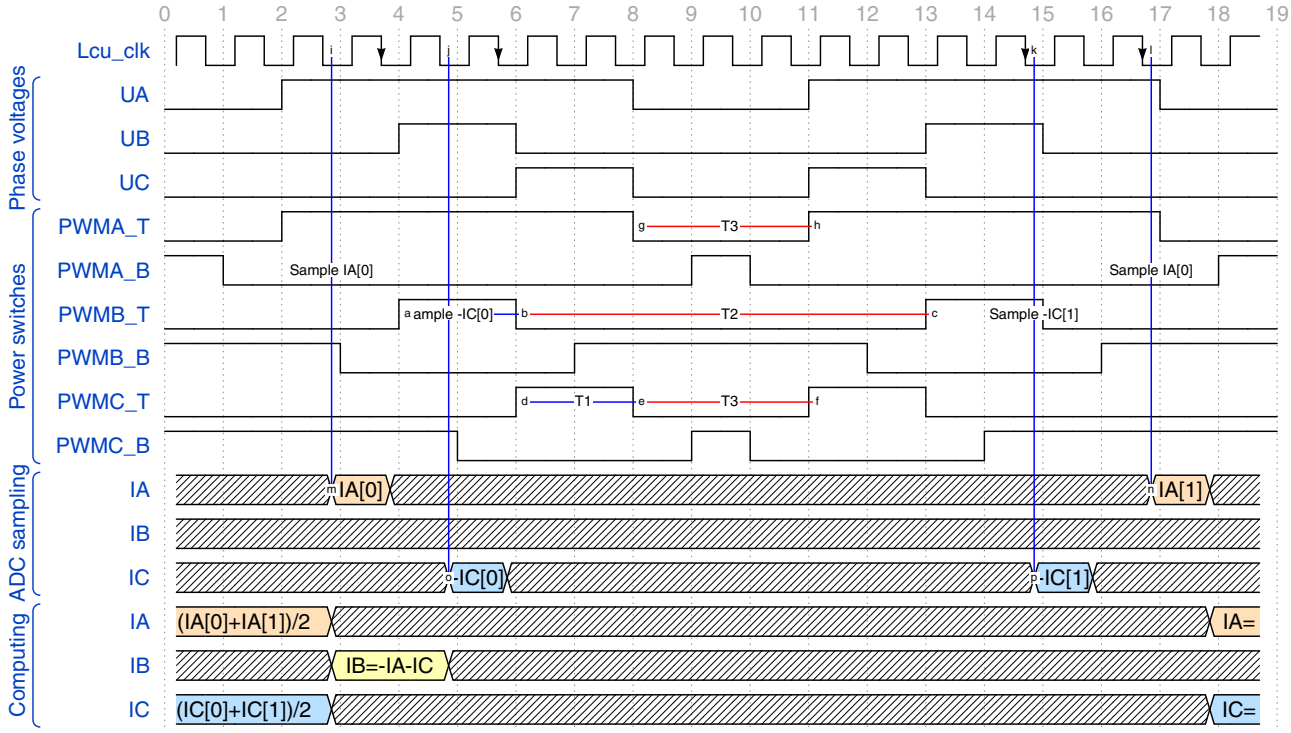


Figure 284. Waveforms for double-switching technique example

59.7.10.8 Implementation for AC motor double-switching controller

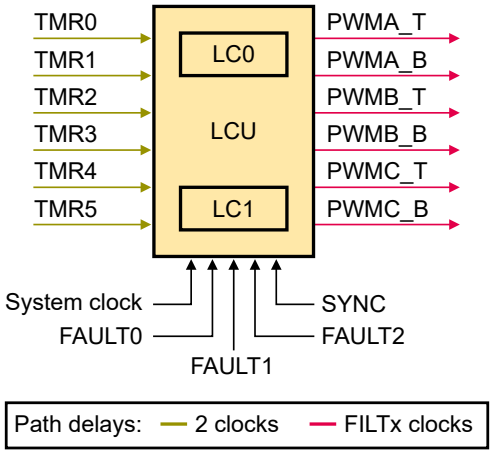


Figure 285. Implementation for AC motor double-switching controller

59.7.10.9 Implementation for controlling one phase

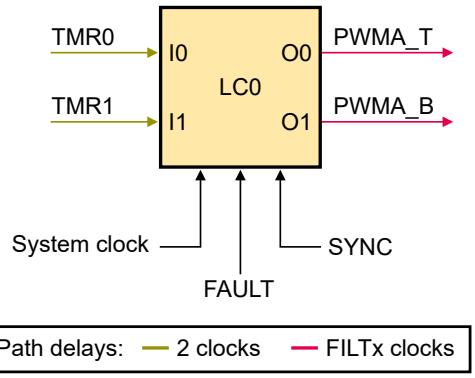


Figure 286. Implementation for controlling one phase

59.7.10.10 Connections and controls

Table 317. Connections and controls

Name	Description
TMR*n*	Synchronized timer input signals
System clock	Clock supplied to the LC
FAULT*n*	Fault signal (see Implementation for AC motor double-switching controller for more details)
SYNC	Sync signal for optional automatic fault recovery
PWM*x*_T	PWM output signal for controlling top power switch
PWM*x*_B	PWM output signal for controlling bottom power switch

59.7.10.11 Truth table

Table 318. Truth table

Inputs		Outputs	
TMR1	TMR0	PWMA_T	PWMB_B
0	0	0	1 delayed by TdH (see Output edge delays)
0	1	1 delayed by TdH (see Output edge delays)	0
1	0	1 delayed by TdH (see Output edge delays)	0
1	1	0	1 delayed by TdH (see Output edge delays)
0	0	0	0
0	1	0	0
1	0	0	0
1	1	0	0

59.7.10.12 Register configuration

Table 319. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	6h	—	—	—
LUTCTRL1	—	—	—	—	—	9h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	1_0000h	—	—	—
FILT1	—	—	—	—	—	1_0000h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	100_0000h			
FCTRL	—	—	—	—	Force inputs 2 and 0 affect output 0, but force inputs 3 and 1 do not affect ouput 0: 101h All four inputs affect output 0: 1111h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	0h	—	—	—	—	—	—
MUXSEL3	0h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	3h			

59.7.10.13 Waveforms for generation of PWM complementary signals using an LC

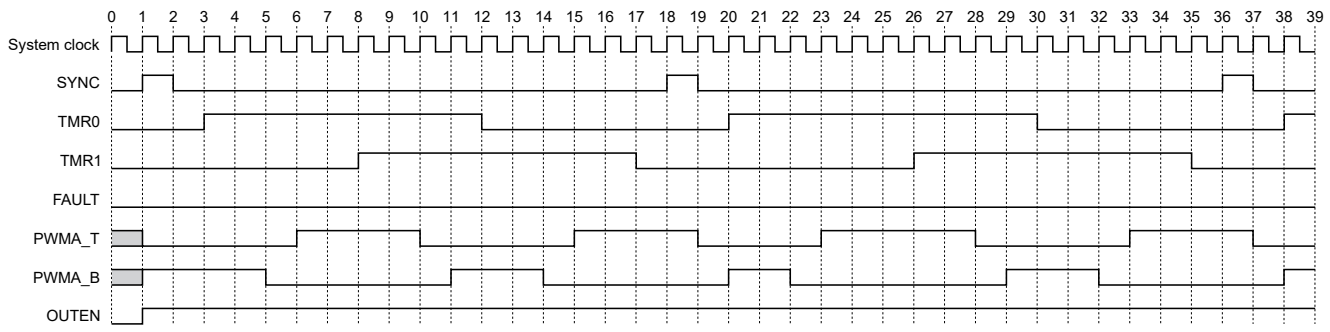


Figure 287. Waveforms for generation of PWM complementary signals using an LC

59.7.10.14 Waveforms for force cleared on force deassertion

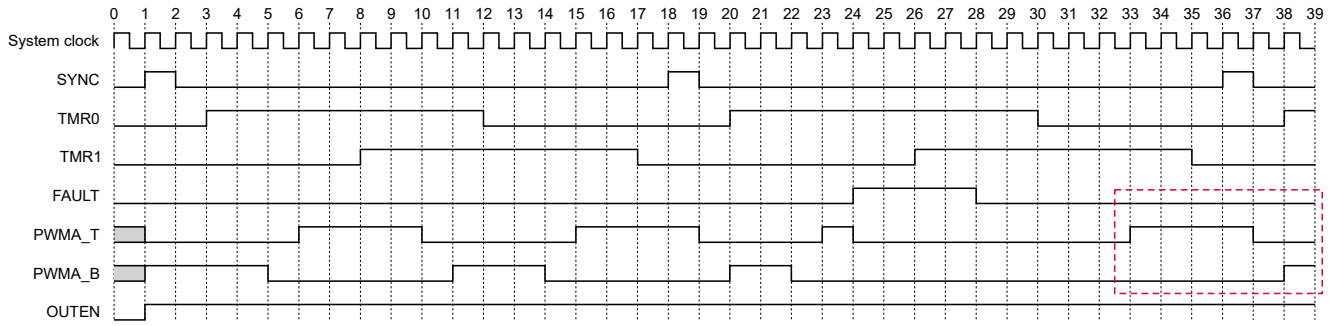


Figure 288. Waveforms for force cleared on force deassertion

59.7.10.15 Waveforms for force cleared on rising sync after force deassertion

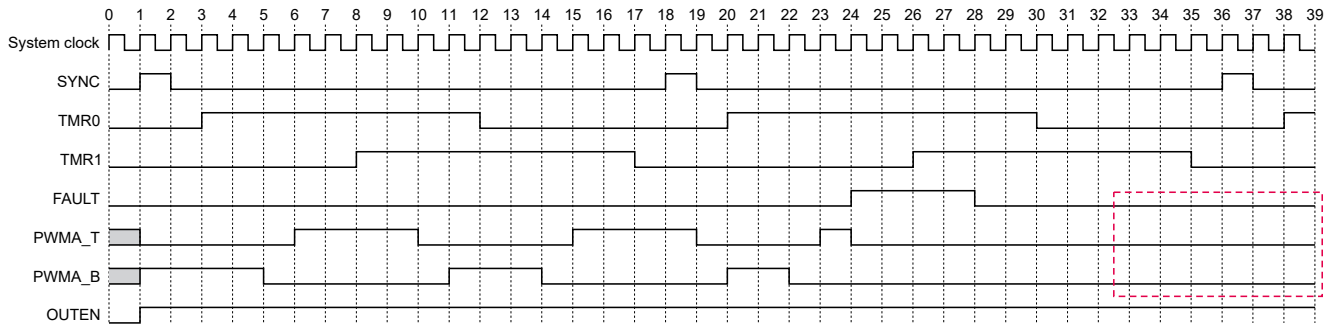


Figure 289. Waveforms for force cleared on rising sync after force deassertion

59.7.11 Brushless DC (BLDC) motor PWM controller

59.7.11.1 Overview

You can use LCU to replace the mechanical commutator in the BLDC motor control application with Hall-sensor position measurement.

In this example implementation, LC0 receives the direction (DIR) and Hall A, Hall B, and Hall C sensor output states, and from those inputs generates the switching states S1, S2, and S3. LC1 and LC2 convert the switching states to PWM or on-off signals for controlling power switches, as illustrated in Waveforms for BLDC motor control technique.

BLDC motor power stage schematic and BLDC motor electrical arrangement illustrate a three-phase BLDC motor power stage with switching pulses and phase currents controlling motor rotation in the CCW direction. The state of the Hall sensors is 010. As the rotor shaft rotates, Hall sensors track the motor position. You accelerate or decelerate the motor by controlling the width of the PWM signal. You can include one or more faults (force inputs) to force power switches instantaneously into the inactive state if there is an overcurrent.

59.7.11.2 Implementation

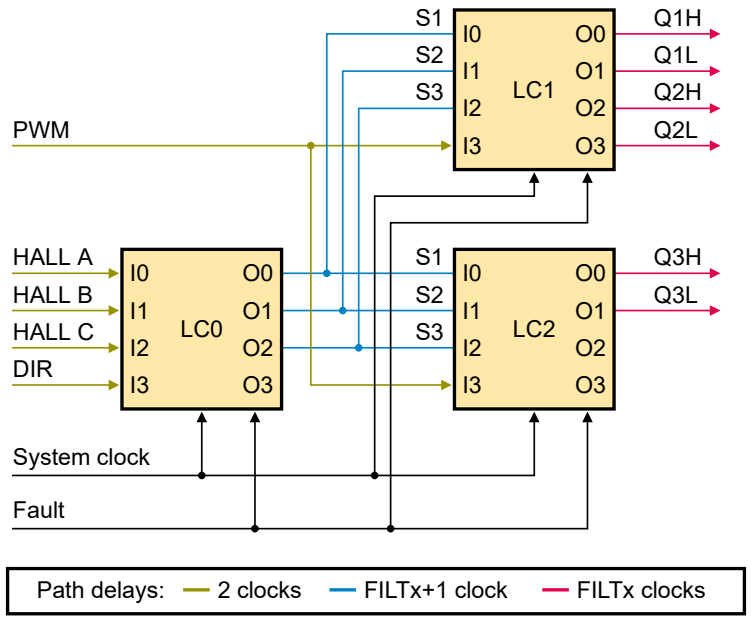


Figure 290. Implementation

59.7.11.3 BLDC motor power stage schematic

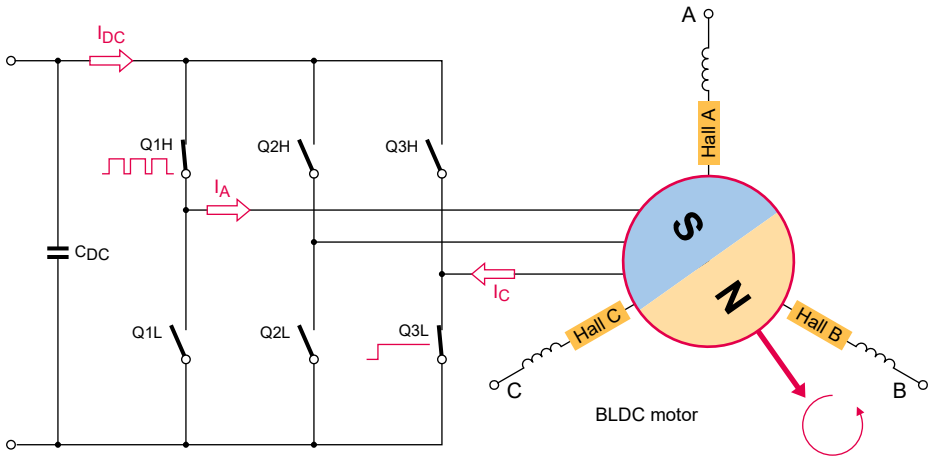


Figure 291. BLDC motor power stage schematic

59.7.11.4 BLDC motor electrical arrangement

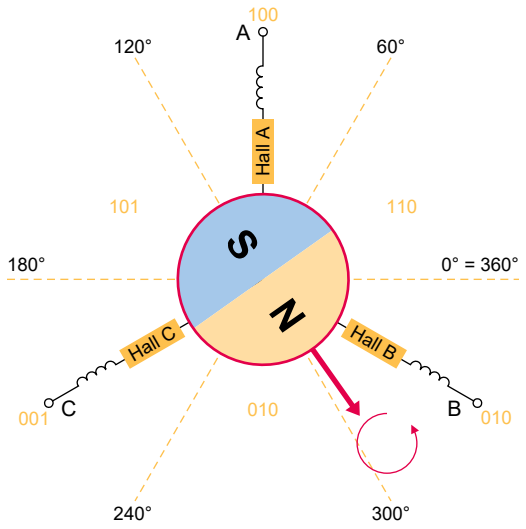


Figure 292. BLDC motor electrical arrangement

59.7.11.5 Waveforms for BLDC motor control technique

This figure illustrates the basic waveforms for controlling a BLDC motor with stator windings and Hall sensors arranged according to BLDC motor electrical arrangement. Configure the external hardware (motor or sensor) so that it can produce these waveforms; then you can configure LCU.

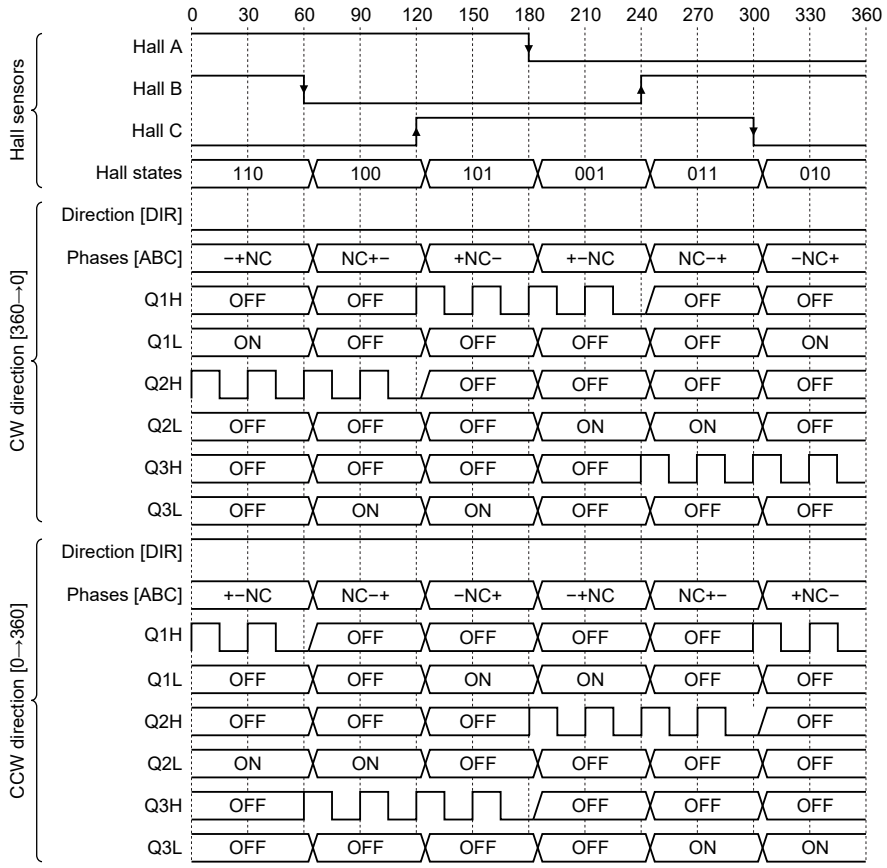


Figure 293. Waveforms for BLDC motor control technique

59.7.11.6 Connections and controls

Table 320. Connections and controls

Input-output group	Name	Description
LC0 inputs	HALL A	Hall sensor A output state
	HALL B	Hall sensor B output state
	HALL C	Hall sensor C output state
	DIR	Direction of shaft rotation, as illustrated in Waveforms for reversing BLDC motor.
LC1 and LC2 inputs	S1	Switching state 1 (output 0 from LC0)
	S2	Switching state 2 (output 1 from LC0)
	S2	Switching state 3 (output 2 from LC0)
	PWM	Output from PWM
LC0, LC1, and LC2 inputs	Clock	LCU clock input
	Fault	Optional fault input

Table continues on the next page...

Table 320. Connections and controls (continued)

Input-output group	Name	Description
LC1 outputs	Q1H	Motor control switch outputs, as illustrated in BLDC motor electrical arrangement
	Q1L	
	Q2H	
	Q2L	
LC2 outputs	Q3H	
	Q3L	

59.7.11.7 Truth table for generating switching states

This table includes all Hall sensor states, even those that never occur because of sensor displacement by 120°.

Table 321. Truth table for generating switching states

Inputs				Outputs			Comments
DIR	Hall C	Hall B	Hall A	S1	S2	S3	
0	0	0	0	0	0	0	Not used
0	0	0	1	1	0	0	Hall states
0	0	1	0	0	1	0	Hall states
0	0	1	1	1	1	0	Hall states
0	1	0	0	0	0	1	Hall states
0	1	0	1	1	0	1	Hall states
0	1	1	0	0	1	1	Hall states
0	1	1	1	0	0	0	Not used
1	0	0	0	0	0	0	Not used
1	0	0	1	0	1	1	Complement of Hall states
1	0	1	0	1	0	1	Complement of Hall states
1	0	1	1	0	0	1	Complement of Hall states
1	1	0	0	1	1	0	Complement of Hall states
1	1	0	1	0	1	0	Complement of Hall states
1	1	1	0	1	0	0	Complement of Hall states
1	1	1	1	0	0	0	Not used

59.7.11.8 Truth table for generating switching pulses for phases A and B

Table 322. Truth table for generating switching pulses for phases A and B

Inputs				Outputs				Comments
PWM	S3	S2	S1	Q1H	Q1L	Q2H	Q2L	
0	0	0	0	0	0	0	0	Not used
0	0	0	1	0	0	0	0	—
0	0	1	0	0	1	0	0	—
0	0	1	1	0	1	0	0	—
0	1	0	0	0	0	0	1	—
0	1	0	1	0	0	0	0	—
0	1	1	0	0	0	0	1	—
0	1	1	1	0	0	0	0	Not used
1	0	0	0	0	0	0	0	Not used
1	0	0	1	0	0	1	0	—
1	0	1	0	0	1	0	0	—
1	0	1	1	0	1	1	0	—
1	1	0	0	1	0	0	1	—
1	1	0	1	1	0	0	0	—
1	1	1	0	0	0	0	1	—
1	1	1	1	0	0	0	0	Not used

59.7.11.9 Truth table for generating switching pulses for phase C

Table 323. Truth table for generating switching pulses for phase C

Inputs				Outputs		Comments
PWM	S3	S2	S1	Q3H	Q3L	
0	0	0	0	0	0	Not used
0	0	0	1	0	1	—
0	0	1	0	0	0	—
0	0	1	1	0	0	—
0	1	0	0	0	0	—
0	1	0	1	0	1	—
0	1	1	0	0	0	—
0	1	1	1	0	0	Not used
1	0	0	0	0	0	Not used
1	0	0	1	0	1	—

Table continues on the next page...

Table 323. Truth table for generating switching pulses for phase C (continued)

Inputs				Outputs		Comments
PWM	S3	S2	S1	Q3H	Q3L	
1	0	1	0	1	0	—
1	0	1	1	0	0	—
1	1	0	0	0	0	—
1	1	0	1	0	1	—
1	1	1	0	1	0	—
1	1	1	1	0	0	Not used

59.7.11.10 Configuration for generating switching states

Table 324. Configuration for generating switching states

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	542Ah	—	—	—
LUTCTRL1	—	—	—	—	—	324Ch	—	—
LUTCTRL2	—	—	—	—	—	—	E70h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	3h	—	—	—	—	—	—
MUXSEL3	4h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	7h			

59.7.11.11 Configuration for switching pulses for phases A and B

Table 325. Configuration for switching pulses for phases A and B

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	3000h	—	—	—
LUTCTRL1	—	—	—	—	—	C0Ch	—	—
LUTCTRL2	—	—	—	—	—	—	A00h	—
LUTCTRL3	—	—	—	—	—	—	—	5050h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	F00_0000h			
FCTRL	—	—	—	—	101_0101h			
SCTRL	—	—	—	—	0h			
MUXSEL4	—	—	—	Dh	—	—	—	—
MUXSEL5	—	—	Eh	—	—	—	—	—
MUXSEL6	—	Fh	—	—	—	—	—	—
MUXSEL7	8h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	F0h			

59.7.11.12 Configuration for switching pulses for phase C

Table 326. Configuration for switching pulses for phase C

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	4400h	—	—	—
LUTCTRL1	—	—	—	—	—	2222h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h

Table continues on the next page...

Table 326. Configuration for switching pulses for phase C (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0300_0000h			
FCTRL	—	—	—	—	101h			
SCTRL	—	—	—	—	0h			
MUXSEL8	—	—	—	Dh	—	—	—	—
MUXSEL9	—	—	Eh	—	—	—	—	—
MUXSEL10	—	Fh	—	—	—	—	—	—
MUXSEL11	8h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	300h			

59.7.11.13 Waveforms for reversing BLDC motor

This figure illustrates the waveforms for controlling a reversing BLDC motor. The DIR input indicates the direction of rotation:

- Logic 0 for CW rotation
- Logic 1 for CCW rotation

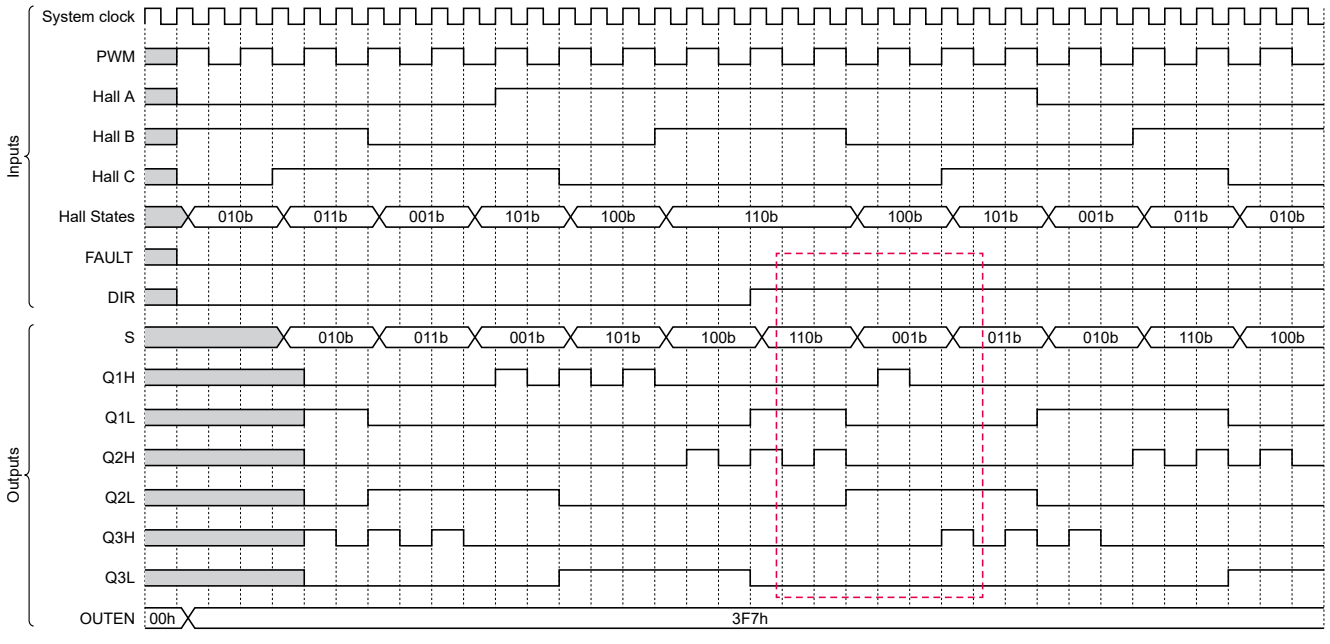


Figure 294. Waveforms for reversing BLDC motor

59.7.11.14 Waveforms for reversing BLDC motor with fault assertion

This figure illustrates the waveforms for the reaction to an external fault in addition to reversing the BLDC motor. On the occurrence of an external fault event, the power switches transition to the inactive state (logic 0). Two clock cycles after external fault deassertion, the power switches start operating normally.

Switching state outputs S1, S2, and S3, generated by LC0, receive an additional delay of one clock cycle on their way to the inputs of other LCs (digital filters bypassed).

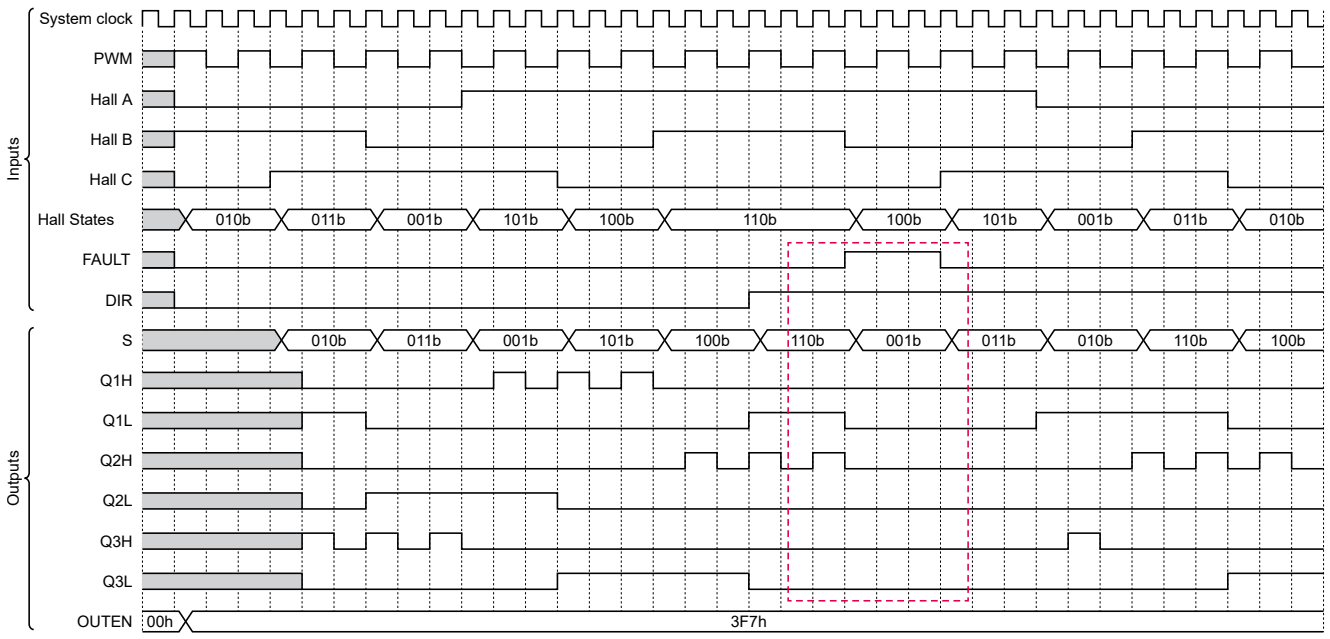


Figure 295. Waveforms for reversing BLDC motor with fault assertion

59.8 Registers

59.8.1 Mapping of 16-bit input, output, and state fields

This table maps inputs, outputs, or states to the bit positions in the following 16-bit fields:

- SWEN[SWEN]
- SWVALUE[SWVALUE]
- OUTEN[OUTEN]
- LCIN[LC_INPUTS]
- SWOUT[SWOUT]
- LCOUT[LCOUT]
- FORCEOUT[FORCEOUT]
- FORCESTS[FORCESTS]
- DBGEN[DBGEN]

LC	Input, output, or state	Bit
0	0	0
	1	1
	2	2
	3	3
1	0	4
	1	5
	2	6
	3	7
2	0	8
	1	9
	2	10
	3	11
3	0	12
	1	13
	2	14
	3	15

59.9 LCU register descriptions

NOTE

Access to address offset 2A4h does not generate a transfer error.

59.9.1 LCU memory map

LCU_0 base address: 4009_8000h

LCU_1 base address: 4009_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	LC 0 Output 0 LUT Control (LC0_LUTCTRL0)	32	RW	0000_0000h
4h	LC 0 Output 1 LUT Control (LC0_LUTCTRL1)	32	RW	0000_0000h
8h	LC 0 Output 2 LUT Control (LC0_LUTCTRL2)	32	RW	0000_0000h
Ch	LC 0 Output 3 LUT Control (LC0_LUTCTRL3)	32	RW	0000_0000h
10h	LC 0 Output 0 Filter (LC0_FILT0)	32	RW	0000_0000h
14h	LC 0 Output 1 Filter (LC0_FILT1)	32	RW	0000_0000h
18h	LC 0 Output 2 Filter (LC0_FILT2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1Ch	LC 0 Output 3 Filter (LC0_FILT3)	32	RW	0000_0000h
20h	LC 0 Interrupt and DMA Enable (LC0_INTDMAEN)	32	RW	0000_0000h
24h	LC 0 Status (LC0_STS)	32	RW	0000_0000h
28h	LC 0 Output Polarity Control (LC0_OUTPOL)	32	RW	0000_0000h
2Ch	LC 0 Force Filter (LC0_FFILT)	32	RW	0000_0000h
30h	LC 0 Force Control (LC0_FCTRL)	32	RW	0000_0000h
34h	LC 0 Sync Control (LC0_SCTRL)	32	RW	0000_0000h
40h	LC 1 Output 0 LUT Control (LC1_LUTCTRL0)	32	RW	0000_0000h
44h	LC 1 Output 1 LUT Control (LC1_LUTCTRL1)	32	RW	0000_0000h
48h	LC 1 Output 2 LUT Control (LC1_LUTCTRL2)	32	RW	0000_0000h
4Ch	LC 1 Output 3 LUT Control (LC1_LUTCTRL3)	32	RW	0000_0000h
50h	LC 1 Output 0 Filter (LC1_FILT0)	32	RW	0000_0000h
54h	LC 1 Output 1 Filter (LC1_FILT1)	32	RW	0000_0000h
58h	LC 1 Output 2 Filter (LC1_FILT2)	32	RW	0000_0000h
5Ch	LC 1 Output 3 Filter (LC1_FILT3)	32	RW	0000_0000h
60h	LC 1 Interrupt and DMA Enable (LC1_INTDMAEN)	32	RW	0000_0000h
64h	LC 1 Status (LC1_STS)	32	RW	0000_0000h
68h	LC 1 Output Polarity Control (LC1_OUTPOL)	32	RW	0000_0000h
6Ch	LC 1 Force Filter (LC1_FFILT)	32	RW	0000_0000h
70h	LC 1 Force Control (LC1_FCTRL)	32	RW	0000_0000h
74h	LC 1 Sync Control (LC1_SCTRL)	32	RW	0000_0000h
80h	LC 2 Output 0 LUT Control (LC2_LUTCTRL0)	32	RW	0000_0000h
84h	LC 2 Output 1 LUT Control (LC2_LUTCTRL1)	32	RW	0000_0000h
88h	LC 2 Output 2 LUT Control (LC2_LUTCTRL2)	32	RW	0000_0000h
8Ch	LC 2 Output 3 LUT Control (LC2_LUTCTRL3)	32	RW	0000_0000h
90h	LC 2 Output 0 Filter (LC2_FILT0)	32	RW	0000_0000h
94h	LC 2 Output 1 Filter (LC2_FILT1)	32	RW	0000_0000h
98h	LC 2 Output 2 Filter (LC2_FILT2)	32	RW	0000_0000h
9Ch	LC 2 Output 3 Filter (LC2_FILT3)	32	RW	0000_0000h
A0h	LC 2 Interrupt and DMA Enable (LC2_INTDMAEN)	32	RW	0000_0000h
A4h	LC 2 Status (LC2_STS)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A8h	LC 2 Output Polarity Control (LC2_OUTPOL)	32	RW	0000_0000h
ACh	LC 2 Force Filter (LC2_FFILT)	32	RW	0000_0000h
B0h	LC 2 Force Control (LC2_FCTRL)	32	RW	0000_0000h
B4h	LC 2 Sync Control (LC2_SCTRL)	32	RW	0000_0000h
200h - 22Ch	Mux Select (MUXSEL0 - MUXSEL11)	32	RW	0000_0000h
280h	Configuration (CFG)	32	RW	0303_0280h
284h	Software Override Enable (SWEN)	32	RW	0000_0000h
288h	Software Override Value (SWVALUE)	32	RW	0000_0000h
28Ch	Output Enable (OUTEN)	32	RW	0000_0000h
290h	Logic Inputs (LCIN)	32	R	0000_0000h
294h	Overridden Inputs (SWOUT)	32	R	0000_0000h
298h	Logic Outputs (LCOUT)	32	R	0000_0000h
29Ch	Forced Outputs (FORCEOUT)	32	R	0000_0000h
2A0h	Force Status (FORCESTS)	32	RW	0000_0000h
2A8h	Debug Mode Enable (DBGEN)	32	RW	0000_0000h

59.9.2 LC n Output m LUT Control (LC0_LUTCTRL0 - LC2_LUTCTRL3)

Offset

For n = 0 to 2; m = 0 to 3:

Register	Offset
LCn_LUTCTRLm	0h + (n × 40h) + (m × 4h)

Function
See LUTCTRL.

Diagram

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

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

Fields

Field	Function
31-16 —	Reserved
15-0 LUTCTRL	LUT Control Specifies the LUT positions, based on the combined LC input value, that result in assertion of this output. For more information, see Logic operations.

59.9.3 LC n Output m Filter (LC0_FILT0 - LC2_FILT3)

Offset

For n = 0 to 2; m = 0 to 3:

Register	Offset
LCn_FILTm	10h + (n × 40h) + (m × 4h)

Function

Specifies the rising- and falling-edge thresholds for the LC output filters.

Diagram

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

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

Fields

Field	Function
31-16 LUT_RISE_FILT	Rise Filter Specifies the number of consecutive clock cycles the filter output must be logic 1 before the output signal goes high. 0000_0000_0000_0000b - Bypass filter All other values - Filter threshold
15-0 LUT_FALL_FILT	Fall Filter Specifies the number of consecutive clock cycles the filter output must be logic 0 before the output signal goes low. 0000_0000_0000_0000b - Bypass filter All other values - Filter threshold

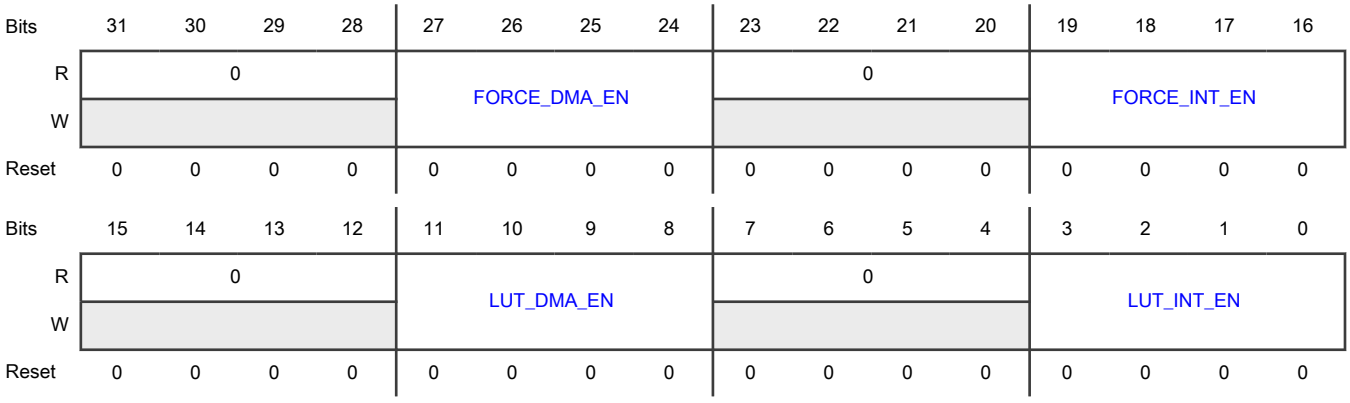
59.9.4 LC n Interrupt and DMA Enable (LC0_INTDMAEN - LC2_INTDMAEN)

Offset

Register	Offset
LC0_INTDMAEN	20h
LC1_INTDMAEN	60h
LC2_INTDMAEN	A0h

Function
Enables interrupt and DMA requests for LUT and force events.

Diagram



Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31-28
—</td><td>Reserved</td></tr>
<tr><td>27-24
FORCE_DMA_EN</td><td>Force DMA Enable
Enables the generation of a DMA request when a force event occurs (LCn_STS[FORCESTS]).
For each bit:
0b - Disable
1b - Enable
Mapping of bits:<table><tr><th>Output</th><th>Register bit</th></tr><tr><td>0</td><td>24</td></tr><tr><td>1</td><td>25</td></tr><tr><td>2</td><td>26</td></tr><tr><td>3</td><td>27</td></tr></table></td></tr>
<tr><td>23-20
—</td><td>Reserved</td></tr>
<tr><td>19-16
FORCE_INT_EN</td><td>Force Interrupt Enable
Enables the generation of an interrupt request when a force event occurs (LCn_STS[FORCESTS]).
For each bit:
0b - Disable
1b - Enable
Mapping of bits:<table><tr><th>Output</th><th>Register bit</th></tr><tr><td>0</td><td>16</td></tr><tr><td>1</td><td>17</td></tr><tr><td>2</td><td>18</td></tr><tr><td>3</td><td>19</td></tr></table></td></tr>
<tr><td>15-12
—</td><td>Reserved</td></tr>
<tr><td>11-8</td><td>LUT DMA Enable</td></tr>
</table>

Table continued from the previous page...

Field	Function
LUT_DMA_EN	Enables the generation of a DMA request when an LUT event occurs (LC*n*_STS[LUT_STS]). For each bit: 0b - Disable 1b - Enable Mapping of bits:

Output	Register bit
0	8
1	9
2	10
3	11

Field	Function
7-4 —	Reserved
3-0 LUT_INT_EN	LUT Interrupt Enable Enables the generation of an interrupt request when an LUT event occurs (LC*n*_STS[LUT_STS]). For each bit: 0b - Disable 1b - Enable Mapping of bits:

Output	Register bit
0	0
1	1
2	2
3	3

59.9.5 LC n Status (LC0_STS - LC2_STS)

Offset

Register	Offset
LC0_STS	24h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
LC1_STS	64h
LC2_STS	A4h

Function

Indicates the occurrence of LUT and force events.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FORCESTS				0				LUT_STS			
W					W1C								W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-8 FORCESTS	Force Event Indicates that a force event has occurred on the associated output. For each bit: 0b - No event 1b - Event occurred Mapping of bits:

Output	Register bit
0	8
1	9
2	10
3	11

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>When you enable DMA for a force output (LCn_INTDMAEN[FORCE_DMA_EN]) and a force event occurs on that output, LCU generates a DMA request. The resulting DMA done signal causes LCU to change the bit to 0, or you can write 1 to the bit.

You can also change these bits to 0 by writing to the corresponding Force Status bits (FORCESTS[FORCESTS]).

The timing of status clearing depends on the selected Force Mode (LCn_FCTRL[FORCE_MODEm]).</td></tr>
<tr><td>7-4

—</td><td>Reserved</td></tr>
<tr><td>3-0

LUT_STS</td><td>LUT Event

Indicates that an LUT event has occurred for the associated LC output.

For each bit:

0b - No event

1b - Event occurred

Mapping of bits:

<table>
<tr><th>Output</th><th>Register bit</th></tr>
<tr><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td></tr>
<tr><td>2</td><td>2</td></tr>
<tr><td>3</td><td>3</td></tr>
</table>

When you enable DMA for an LUT event on an output (LCn_INTDMAEN[LUT_DMA_EN]) and an LUT event occurs on that output, LCU generates a DMA request. The resulting DMA done signal causes LCU to change the bit to 0.

When you enable interrupt request for an output (LCn_INTDMAEN[LUT_INT_EN]) and an LUT event occurs on that output, LCU generates an interrupt request. Write 1 to the bit to change it to 0.</td></tr>
</table>

59.9.6 LC n Output Polarity Control (LC0_OUTPOL - LC2_OUTPOL)

Offset

Register	Offset
LC0_OUTPOL	28h
LC1_OUTPOL	68h
LC2_OUTPOL	A8h

Function

Specifies the polarity of the LC outputs.

Diagram

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

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

Fields

Field	Function
31-4 —	Reserved
3-0 OUTPOL	Output Polarity Specifies the polarity of the outputs. For each bit: 0b - Not inverted 1b - Inverted Mapping of bits:

Output	Bit
0	0
1	1
2	2
3	3

59.9.7 LC n Force Filter (LC0_FFILT - LC2_FFILT)

Offset

Register	Offset
LC0_FFILT	2Ch

Table continues on the next page...

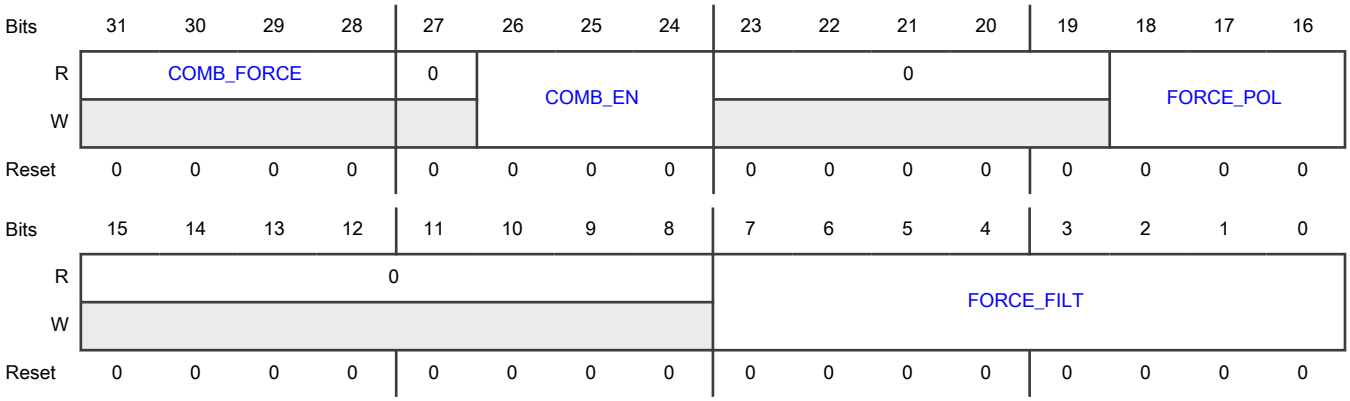
Table continued from the previous page...

Register	Offset
LC1_FFILT	6Ch
LC2_FFILT	ACh

Function

Controls the force filter for this LC.

Diagram



Fields

Field	Function
31-28 COMB_FORCE	Combined Sensed Force Input Indicates the combined value of force inputs to each output. For each bit: 0b - Logic low 1b - Logic high Mapping of bits:

Output	Bit
0	28
1	29
2	30
3	31

27 —	Reserved

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>26-24
COMB_EN</td><td>Combinational Force Path (CFP) Enable
Enables an active force input to combinationally affect the LC outputs. When CFP is not enabled, force inputs must be synchronized and then optionally filtered. When you enable CFP (write 1 to one of these bits), the corresponding force input bypasses synchronization and filtering and immediately affects the LC output.
The Force Filter (FORCE_FILT) is still in effect. Force inputs that do not meet the pulse width requirements for synchronization and filtering are not registered in Force Status (FORCESTS) and you do not need to write 1 to change the bit to 0.
CFP provides a safety factor when the LC outputs drive a motor and immediate response is required. In some topologies, an oscillation can result if the assertion of the force input causes the LC output to turn off. This situation in turn leads to deassertion of the force input and the output turning back on. In general, you must disable CFP unless the LCU outputs drive a motor or similar safety-critical application, which requires an immediate response without synchronization delays.</td></tr>
<tr><td>23-19
—</td><td>Reserved</td></tr>
<tr><td>18-16
FORCE_POL</td><td>Force Input Polarity
Specifies the polarity of the force inputs to this LC.
For each bit:
0b - Not inverted
1b - Inverted
Mapping of bits:<table><tr><th>Input</th><th>Bit</th></tr><tr><td>0</td><td>16</td></tr><tr><td>1</td><td>17</td></tr><tr><td>2</td><td>18</td></tr><tr><td>3</td><td>19</td></tr></table></td></tr>
<tr><td>15-8
—</td><td>Reserved</td></tr>
<tr><td>7-0
FORCE_FILT</td><td>Force Filter
Specifies the count, in clock cycles, that a force input must remain at a given logic state before the filtered force input switches state.
0000_0000b - Bypass filter
All other values - Filter threshold</td></tr>
</table>

59.9.8 LC n Force Control (LC0_FCTRL - LC2_FCTRL)

Offset

Register	Offset
LC0_FCTRL	30h
LC1_FCTRL	70h
LC2_FCTRL	B0h

Function

Provides control of the force inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SYNC_SEL3		FORCE_MODE 3		FORCE_SENSE3				SYNC_SEL2		FORCE_MODE 2		FORCE_SENSE2			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SYNC_SEL1		FORCE_MODE 1		FORCE_SENSE1				SYNC_SEL0		FORCE_MODE 0		FORCE_SENSE0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 SYNC_SEL3	Sync Select Selects which sync input to use for output 3 of this LC. 00b - Sync input 0 01b - Sync input 1 10b - Reserved 11b - Reserved
29-28 FORCE_MODE 3	Force Clearing Mode Specifies the timing for clearing force events for output 3 in this LC. 00b - Deassertion. Cleared on deassertion of force inputs 01b - Rising sync after deassertion. Cleared on rising sync after deassertion of force inputs 10b - Writing 1 after deassertion. Cleared by writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits after deassertion of force inputs

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>11b - Rising sync after writing 1 and deassertion. Cleared on rising sync after writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits and deassertion of force inputs</td></tr>
<tr><td>27-24
FORCE_SENSE3</td><td>Force Input Sensitivity
Selects which force inputs affect output 3 of this LC.
For each bit:
0b - Does not affect
1b - Affects
Mapping of bits:
<table><tr><th>Input</th><th>Bit</th></tr><tr><td>0</td><td>24</td></tr><tr><td>1</td><td>25</td></tr><tr><td>2</td><td>26</td></tr><tr><td>Reserved</td><td>27</td></tr></table>
Example: 011b selects force inputs 0 and 1, but not 2.</td></tr>
<tr><td>23-22
SYNC_SEL2</td><td>Sync Select
Selects which sync input to use for output 2 of this LC.
00b - Sync input 0
01b - Sync input 1
10b - Reserved
11b - Reserved</td></tr>
<tr><td>21-20
FORCE_MODE2</td><td>Force Clearing Mode
Specifies the timing for clearing force events for output 2 in this LC.
00b - Deassertion. Cleared on deassertion of force inputs
01b - Rising sync after deassertion. Cleared on rising sync after deassertion of force inputs
10b - Writing 1 after deassertion. Cleared by writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits after deassertion of force inputs
11b - Rising sync after writing 1 and deassertion. Cleared on rising sync after writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits and deassertion of force inputs</td></tr>
<tr><td>19-16
FORCE_SENSE2</td><td>Force Input Sensitivity
Selects which force inputs affect output 2 of this LC.
For each bit:</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - Does not affect
1b - Affects
Mapping of bits:
<table><tr><th>Input</th><th>Bit</th></tr><tr><td>0</td><td>16</td></tr><tr><td>1</td><td>17</td></tr><tr><td>2</td><td>18</td></tr><tr><td>Reserved</td><td>19</td></tr></table>
Example: 011b selects force inputs 0 and 1, but not 2.</td></tr>
<tr><td>15-14
SYNC_SEL1</td><td>Sync Select
Selects which sync input to use for output 1 of this LC.
00b - Sync input 0
01b - Sync input 1
10b - Reserved
11b - Reserved</td></tr>
<tr><td>13-12
FORCE_MODE1</td><td>Force Clearing Mode
Specifies the timing for clearing force events for output 1 in this LC.
00b - Deassertion. Cleared on deassertion of force inputs
01b - Rising sync after deassertion. Cleared on rising sync after deassertion of force inputs
10b - Writing 1 after deassertion. Cleared by writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits after deassertion of force inputs
11b - Rising sync after writing 1 and deassertion. Cleared on rising sync after writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits and deassertion of force inputs</td></tr>
<tr><td>11-8
FORCE_SENSE1</td><td>Force Input Sensitivity
Selects which force inputs affect output 1 of this LC.
For each bit:
0b - Does not affect
1b - Affects
Mapping of bits:
<table><tr><th>Input</th><th>Bit</th></tr><tr><td>0</td><td>8</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Input</th><th>Bit</th></tr>
<tr><td>1</td><td>9</td></tr>
<tr><td>2</td><td>10</td></tr>
<tr><td>Reserved</td><td>11</td></tr>
</table>
Example: 011b selects force inputs 0 and 1, but not 2.
</td></tr>
<tr><td>7-6
SYNC_SEL0</td><td>Sync Select
Selects which sync input to use for output 0 of this LC.
00b - Sync input 0
01b - Sync input 1
10b - Reserved
11b - Reserved</td></tr>
<tr><td>5-4
FORCE_MODE0</td><td>Force Clearing Mode
Specifies the timing for clearing force events for output 0 in this LC.
00b - Deassertion. Cleared on deassertion of force inputs
01b - Rising sync after deassertion. Cleared on rising sync after deassertion of force inputs
10b - Writing 1 after deassertion. Cleared by writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits after deassertion of force inputs
11b - Rising sync after writing 1 and deassertion. Cleared on rising sync after writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits and deassertion of force inputs</td></tr>
<tr><td>3-0
FORCE_SENSE0</td><td>Force Input Sensitivity
Selects which force inputs affect output 0 of this LC.
For each bit:
0b - Does not affect
1b - Affects
Mapping of bits:
<table>
<tr><th>Input</th><th>Bit</th></tr>
<tr><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td></tr>
<tr><td>2</td><td>2</td></tr>
<tr><td>Reserved</td><td>3</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Input</th><th>Bit</th></tr><tr><td colspan="2">Example: 011b selects force inputs 0 and 1, but not 2.</td></tr></table>

59.9.9 LC n Sync Control (LC0_SCTRL - LC2_SCTRL)

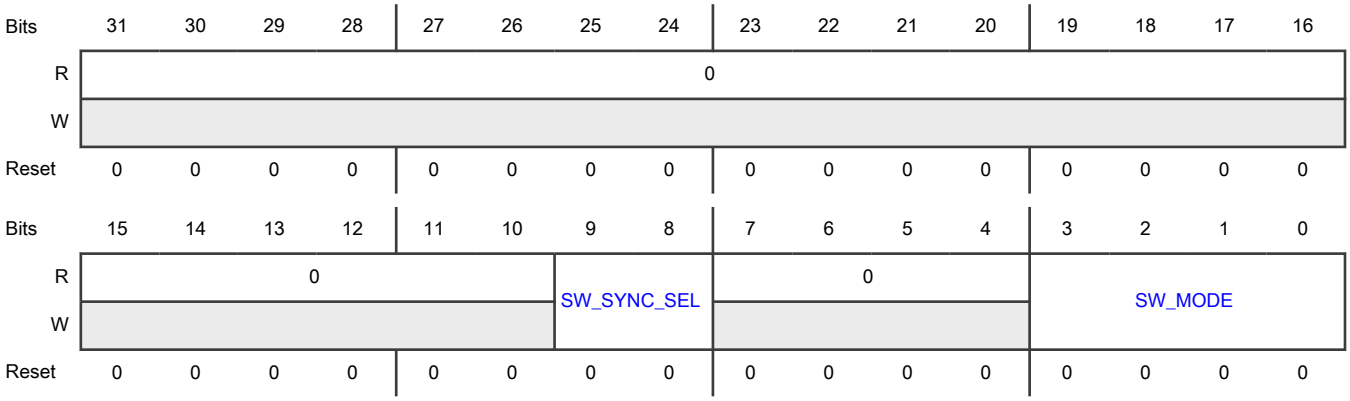
Offset

Register	Offset
LC0_SCTRL	34h
LC1_SCTRL	74h
LC2_SCTRL	B4h

Function

Controls the software sync behavior.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-8 SW_SYNC_SEL	Software Sync Select Selects which sync input to use for software synced mode. 00b - Sync input 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Sync input 1 10b - Reserved 11b - Reserved
7-4 —	Reserved
3-0 SW_MODE	Software Sync Mode Specifies the software sync mode for the inputs to this LC. When Software Override is enabled (SWEN), these bits control whether Software Override Value (SWVALUE) changes occur immediately or on the rising edge of the selected sync pulse. For each bit: 0b - Immediate 1b - On rising edge of sync Mapping of bits:

Input	Bit
0	0
1	1
Reserved	2
Reserved	3

59.9.10 Mux Select (MUXSEL0 - MUXSEL11)

Offset

For a = 0 to 11:

Register	Offset
MUXSELa	200h + (a × 4h)

Function

Selects the sources for inputs to the LCs.

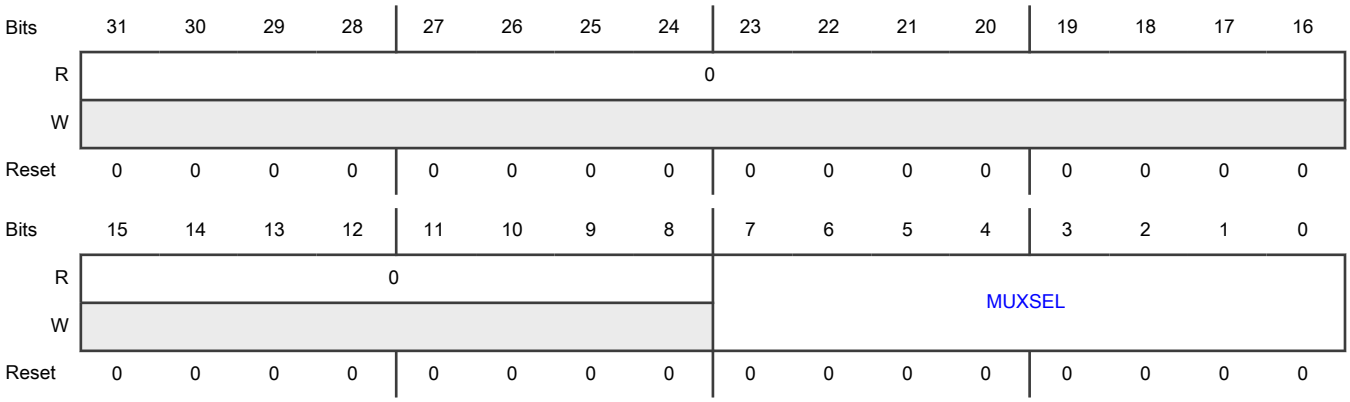
Register	LC	Input
MUXSEL0	0	0
MUXSEL1		1

Table continues on the next page...

Table continued from the previous page...

Register	LC	Input
MUXSEL2		2
MUXSEL3		3
MUXSEL4	1	0
MUXSEL5		1
MUXSEL6		2
MUXSEL7		3
MUXSEL8	2	0
MUXSEL9		1
MUXSEL10		2
MUXSEL11		3

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 MUXSEL	Mux Select Selects the source of the LC input. All LU_IN inputs go through a two-stage synchronizer, but the LU_OUT signals fed back to this mux do not have this two-stage delay because they are already synchronous to LCU clock. 0000_0000b - Logic 0 0000_0001b-0000_1100b - LU_IN0 to LU_IN11 0000_1101b-0001_1000b - LU_OUT0 to LU_OUT11 All other values are reserved.

59.9.11 Configuration (CFG)

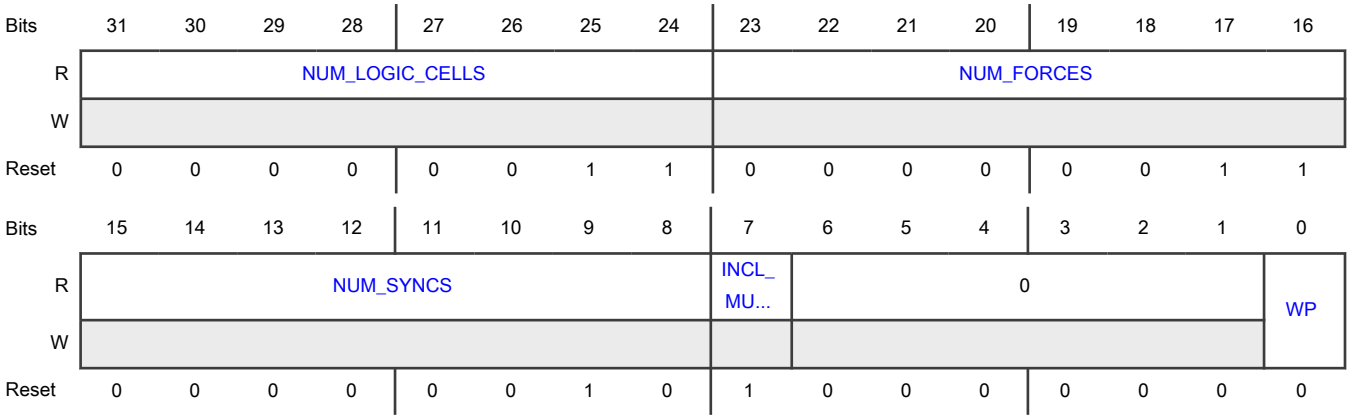
Offset

Register	Offset
CFG	280h

Function

Indicates the LCU configuration, and provides write protection.

Diagram



Fields

Field	Function
31-24 NUM_LOGIC_CELLS	LCs Indicates the number of LCs.
23-16 NUM_FORCES	Force Inputs Indicates the number of force inputs for each LC.
15-8 NUM_SYNCS	Sync Inputs Indicates the number of sync inputs for each LC.
7 INCL_MUXES	Input Muxing Indicates whether LCU supports input muxing. 0b - Not supported 1b - Supported
6-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 WP	Write Protect Turns on write protection for all LCU registers except SWVALUE, LC*n*_STS, and FORCESTS. **NOTE** You can only write 1 to this field; you cannot turn off write protection after you turn it on.

59.9.12 Software Override Enable (SWEN)

Offset

Register	Offset
SWEN	284h

Function

Enables overrides for LC inputs.

Diagram

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

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

Fields

Field	Function
31-12 —	Reserved
11-0 SWEN	Software Override Enable Enables software override of LC inputs. For each bit:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable Mapping of bits: See Mapping of 16-bit input, output, and state fields.

59.9.13 Software Override Value (SWVALUE)

Offset

Register	Offset
SWVALUE	288h

Function

Specifies the software override value for each LC input.

Diagram

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

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

Fields

Field	Function
31-12 —	Reserved
11-0 SWVALUE	Software Override Value Specifies the software override value for each LC input. For each bit: 0b - 0 1b - 1 Mapping of bits: See Mapping of 16-bit input, output, and state fields.

59.9.14 Output Enable (OUTEN)

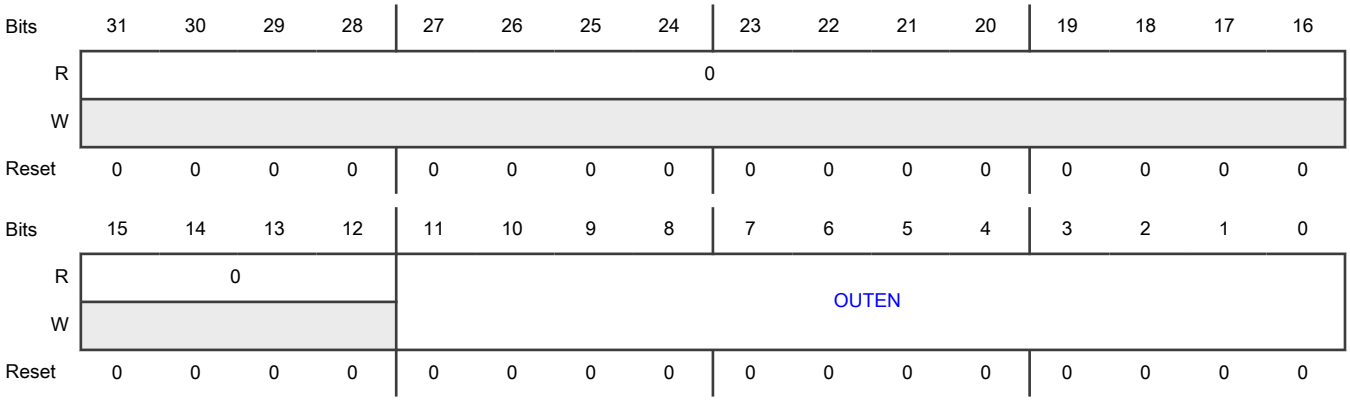
Offset

Register	Offset
OUTEN	28Ch

Function

Enables LC outputs.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 OUTEN	Output Enables Enables LC outputs. For each bit: 0b - Disable 1b - Enable Mapping of bits: See Mapping of 16-bit input, output, and state fields.

59.9.15 Logic Inputs (LCIN)

Offset

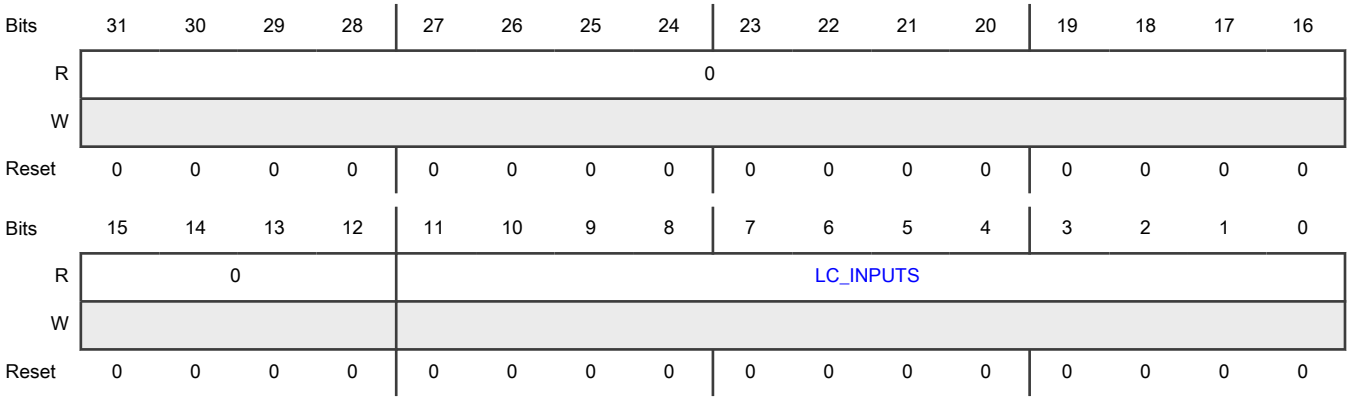
Register	Offset
LCIN	290h

Function

Indicates states of LC inputs.

If you write to this register, LCU generates a bus transfer error.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 LC_INPUTS	Logic Inputs Indicates states of LC inputs. For each bit: 0b - 0 1b - 1 Mapping of bits: See Mapping of 16-bit input, output, and state fields.

59.9.16 Overridden Inputs (SWOUT)

Offset

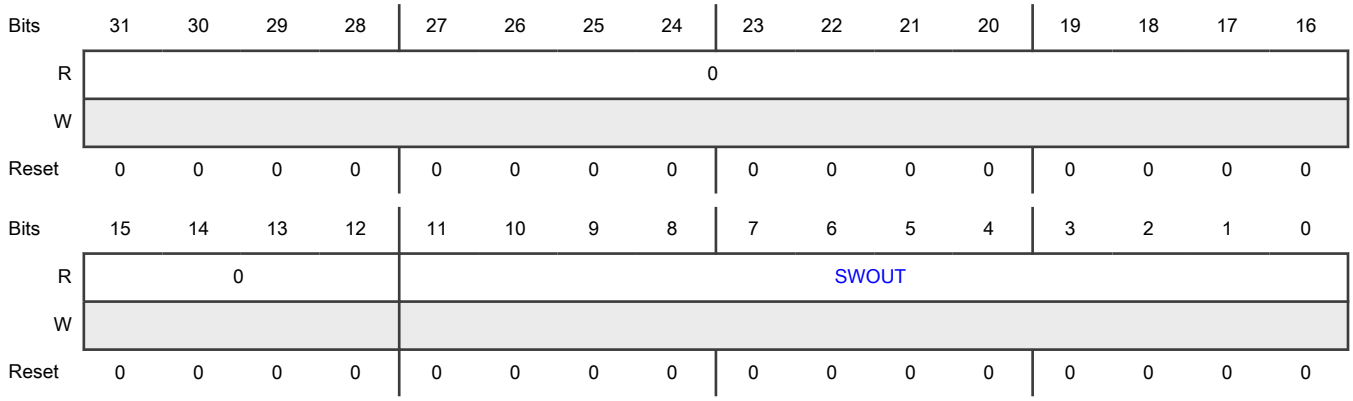
Register	Offset
SWOUT	294h

Function

Indicates states of LC inputs or states of software-overridden inputs, depending upon the state of the corresponding SWEN bit.

If you write to this register, LCU generates a bus transfer error.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 SWOUT	Overridden Inputs Indicates states of LC inputs or software-overridden inputs. For each bit, when the corresponding SWEN bit is 0: 0b - LC input is 0 1b - LC input is 1 For each bit, when the corresponding SWEN bit is 1: 0b - Software-overridden input is 0 1b - Software-overridden input is 1 Mapping of bits: See Mapping of 16-bit input, output, and state fields.

59.9.17 Logic Outputs (LCOUT)

Offset

Register	Offset
LCOUT	298h

Function

Indicates states of LC outputs.

If you write to this register, LCU generates a bus transfer error.

Diagram

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

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

Fields

Field	Function
31-12 —	Reserved
11-0 LCOUT	Logic Outputs Indicates states of LC outputs. For each bit: 0b - 0 1b - 1 Mapping of bits: See Mapping of 16-bit input, output, and state fields.

59.9.18 Forced Outputs (FORCEOUT)

Offset

Register	Offset
FORCEOUT	29Ch

Function

Indicates the current state of the force outputs for all LCU logic outputs. Writing to this register generates a bus transfer error.

Diagram

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

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

Fields

Field	Function
31-12 —	Reserved
11-0 FORCEOUT	Forced Outputs Indicates the current state of force outputs for the logic outputs. For each bit: 0b - Not asserted 1b - Asserted Mapping of bits: See Mapping of 16-bit input, output, and state fields.

59.9.19 Force Status (FORCESTS)

Offset

Register	Offset
FORCESTS	2A0h

Function

See FORCESTS.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FORCESTS											
W					W1C											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 FORCESTS	Force Status Indicates the current force states of all LCs, mirrored from LC*n*_STS[FORCESTS]. Change these bits to 0 by writing 1 to them or to the corresponding bit in the specific LC*n*_STS register. This field allows you to simultaneously change FORCESTS bits to 0 across multiple LCs. For each bit: 0b - Not in force state 1b - Read: In force state — Write: Clear force state bit Mapping of bits: See Mapping of 16-bit input, output, and state fields.

59.9.20 Debug Mode Enable (DBGEN)

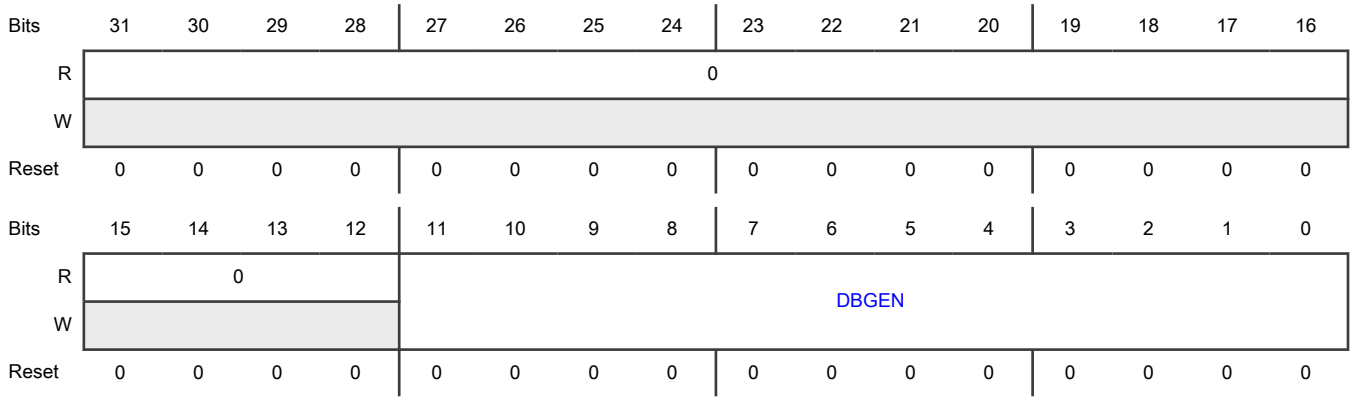
Offset

Register	Offset
DBGEN	2A8h

Function

Enables outputs to continue operation when the chip is in Debug mode.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 DBGEN	Debug Mode Enable Enables outputs to continue operation in Debug mode. For each bit: 0b - Inactive 1b - Continue normal operation Mapping of bits: See Mapping of 16-bit input, output, and state fields.

59.10 Glossary

CCW	Counterclockwise
CW	Clockwise
LC	Logic cell
LUT	Lookup table

Chapter 60
Enhanced Modular IO Subsystem (eMIOS)

60.1 Chip-specific eMIOS information

60.1.1 eMIOS configuration

This chip has up to three instances of eMIOS that are each configured as shown in the following tables.

Table 327. eMIOS instances

Instance	MWCT2D17S	MWCT2015S/ MWCT2016S/MWCT2D16S
eMIOS_0	Yes	Yes
eMIOS_1	Yes	Yes
eMIOS_2	Yes	No

Table 328. eMIOS configuration

Parameter	eMIOS_0	eMIOS_1	eMIOS_2
Timer width	16 bits	16 bits	16 bits
Number of channels	24	24	24
Channels configuration	See EMIOS channel configuration figure below		
Local Channel prescaler width	4	4	4
Number of global counter buses	2	2	2
Number of global prescaler	1	1	1
Global channel prescaler width	8	8	8

NOTE

eMIOS is clocked by CORE_CLK.

60.1.2 Channel types

The eMIOS contain 4 different types of channel, which are listed below.

Table 329. eMIOS channel types

Mode Description	Name	Ch TypeX	Ch TypeY	Ch TypeG	Ch TypeH
General Purpose Input / Output	GPIO	X	X	X	X
Single Action Input Capture	SAIC	X	X	X	X

Table continues on the next page...

Table 329. eMIOS channel types (continued)

Mode Description	Name	Ch TypeX	Ch TypeY	Ch TypeG	Ch TypeH
Single Action Output Compare	SAOC	X	X	X	X
Modulus Counter	MC	X			
Modulus Counter Buffered (Up / Down)	MCB	X		X	
Input Pulse Width Measurement	IPWM			X	X
Input Period Measurement	IPM			X	X
Double Action Output Compare	DAOC			X	X
Output Pulse Width and Frequency Modulation Buffered	OPWFMB	X		X	
Center aligned Output PWM Buffered with dead time	OPWMCB			X	
Output Pulse Width Modulation Buffered	OPWMB	X	X	X	X
Output Pulse Width Modulation Trigger	OPWMT	X	X	X	X
Pulse Edge Counting	PEC			X	

NOTE

ChType X and G has internal counters.

Table 330. eMIOS channel configuration

Channel number	eMIOS0	eMIOS1	eMIOS2
CH0	X	X	X
CH8	X	X	X
CH16	X	X	X
CH22	X	X	X
CH23	X	X	X
CH1	G	H	H
CH2	G	H	H

Table continues on the next page...

Table 330. eMIOS channel configuration (continued)

CH3	G	H	H
CH4	G	H	H
CH5	G	H	H
CH6	G	H	H
CH7	G	H	H
CH9	H	H	H
CH10	H	H	H
CH11	H	H	H
CH12	H	H	H
CH13	H	H	H
CH14	H	H	H
CH15	H	H	H
CH17	Y	Y	Y
CH18	Y	Y	Y
CH19	Y	Y	Y
CH20	Y	Y	Y
CH21	Y	Y	Y

60.1.3 Channel configuration

The eMIOS channel configuration with respect to Type and Counter_bus is shown below.

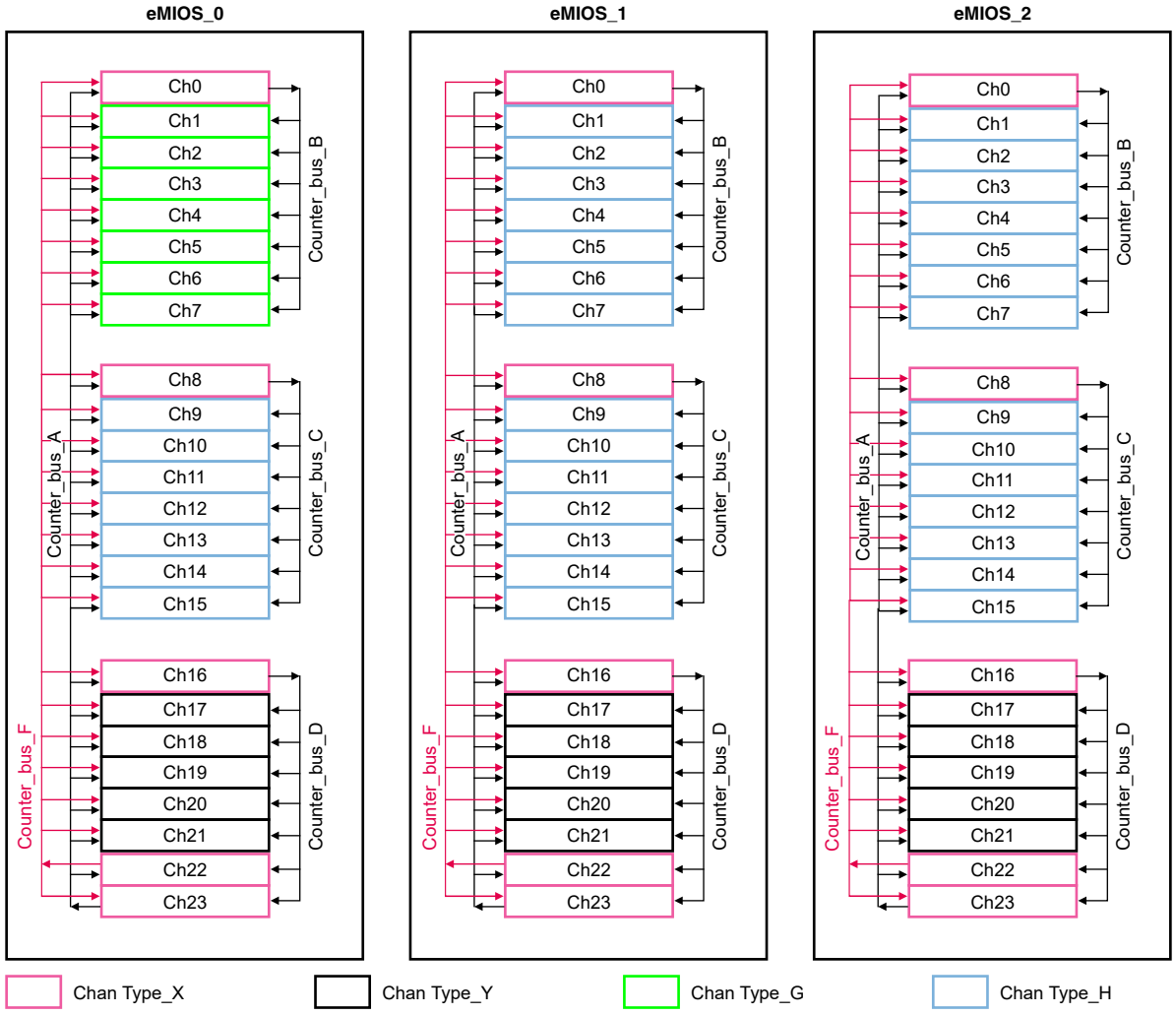


Figure 296. EMIOS channel types

60.1.4 eMIOS disabling

eMIOS can disable it's output using it's disable inputs. Disable inputs[3:0] are driven from the flag out bits[11:8].

Disable inputs	
eMIOS0 Output disable input[3:0]	trgmux_output[31:28]
eMIOS1 Output disable input[3:0]	trgmux_output[31:28]
eMIOS2 Output disable input[3:0]	trgmux_output[31:28]

For details, see the TRGMUX connectivity file attached to this document.

60.1.5 BCTU Interface

Each eMIOS channel, with the exception of the main counterbus channel (ch[23]) is capable of triggering the BCTU. See BCTU chapter for mapping. BCTU allows triggering from as many eMIOS modes as possible.

60.3 Introduction

eMIOS provides independent channels, UCs, that you configure to generate or measure time events for different functions in different chip applications.

eMIOS distributes these channels across a number of global and local counter buses. Each local bus is dedicated to a group of eight contiguous channels. Each channel can generate its own timebase, and each counter bus has its timebase provided by a dedicated channel. For a list of counter buses with their assigned channels and timebase sources, see Counter buses, channels, and timebase sources.

60.4 Block diagram

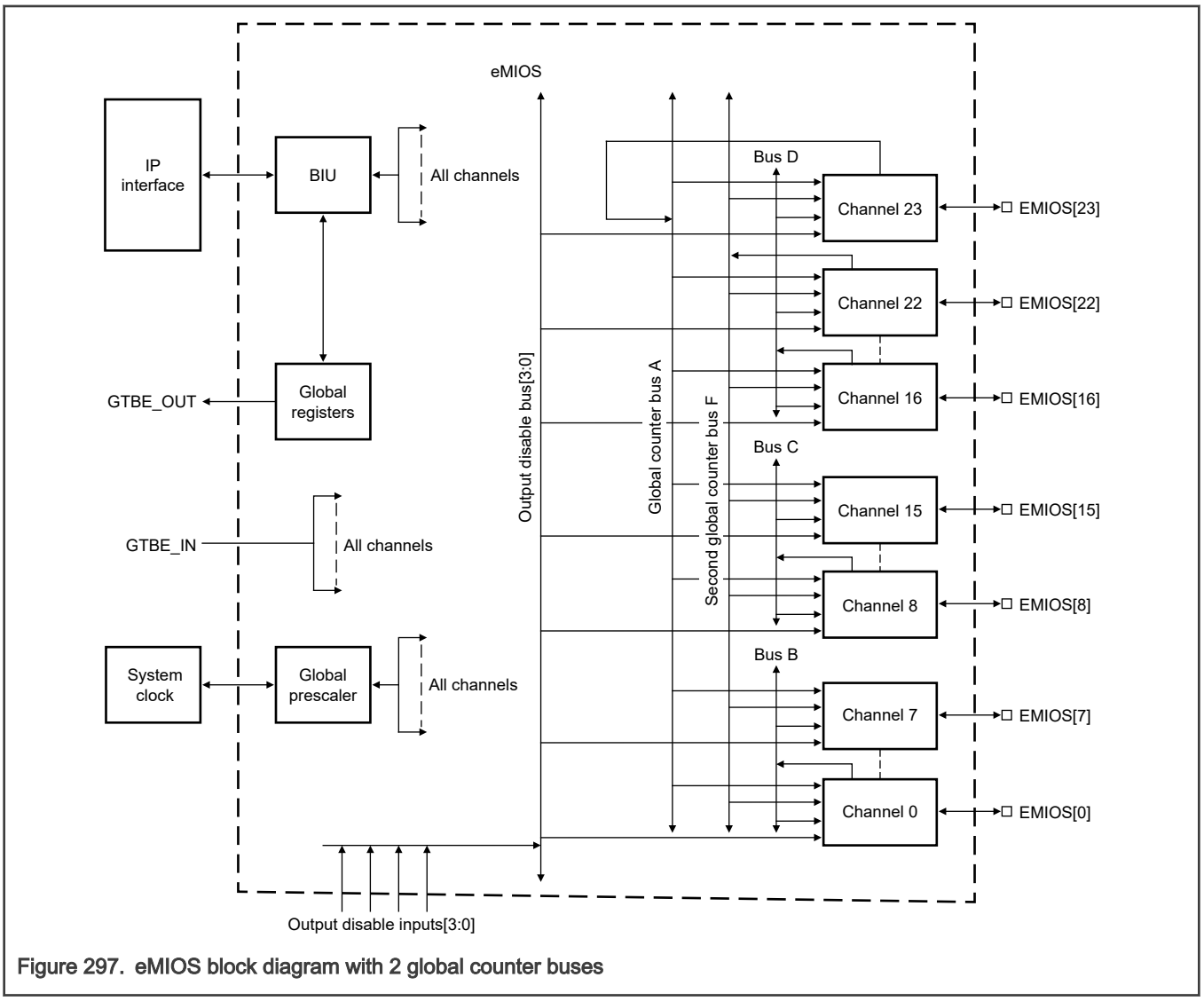


Figure 297. eMIOS block diagram with 2 global counter buses

60.5 Features

- 24 UCs distributed across local counter buses as shown in Counter buses, channels, and timebase sources
- Global counter bus A driven by UC23
- Global counter bus F driven by UC22
- Synchronized timebases shared through the counter buses

- Dedicated timebase for each channel, distinct from the counter buses
- Global clock prescaler (GCP)
- One CP per channel
- Dedicated set of control and status registers for each UC
- 16-bit–wide data registers
- Shadow flag register (GFLAG) to access all channel flags with one read access
- Ability to freeze the UC state for debug purposes
- Motor control capability

60.6 Functional description

60.6.1 Reset

eMIOS resets asynchronously. Reset affects all registers.

On reset, UCs enter GPIO input mode.

60.6.2 Counter buses, channels, and timebase sources

Table 331. Counter buses, channels, and timebase sources

Counter bus	Channels	Timebase channel
Global bus A	All UCs	23
Local bus B	• UC0 • UC1 • UC2 • UC3 • UC4 • UC5 • UC6 • UC7	0
Local bus C	• UC8 • UC9 • UC10 • UC11 • UC12 • UC13 • UC14 • UC15	8
Local bus D	• UC16	16

Table continues on the next page...

Table 331. Counter buses, channels, and timebase sources (continued)

Counter bus	Channels	Timebase channel
	• UC17 • UC18 • UC19 • UC10 • UC21 • UC22 • UC23	
Global bus F	All UCs	22

60.6.3 Unified channels (UC)

60.6.3.1 Overview

Each UC contains the following:

- Two double-buffered data registers, A*n* and B*n*, that allow up to two events—input capture, output compare, or both—to occur before software intervention is needed
- Two comparators, A and B, that indicate when the selected counter bus is equal to the value in A*n* and B*n*
- An internal counter (CNT*n*[C]) that runs in all modes except GPIO.

  You can use this counter as a local timebase or to count input events. You can use it as a timebase if CNT*n*[C] is not used in the current mode.
- An output flip-flop that holds the logic level to be applied to the output pin
- A status register, UC Status n (S0 - S23), that flags input capture and match events, indicates flag overruns and overflows, and shows the input and output pin states
- A control register, UC Control n (C0 - C23), that controls UC operation

You control the following operational characteristics for each UC:

- The logic (C*n*[MODE]) that specifies the behavior of the UC (see UC modes)
- The counter bus (C*n*[BSL]) that provides the timebase for all timing functions
- The CP (C*n*[UCPRE] and C2_*n*[UCEXTPRE])
- The minimum input pulse width (C*n*[IF]) for the input filter that determines valid pin transitions
- Which input edges to detect (C*n*[EDSEL])
- Which of four output disable input signals (C*n*[ODISSL]) to use for disabling outputs
- For all output modes except GPIO, whether to disable the output flip-flop (C*n*[ODIS])

UC block diagram illustrates the basic UC architecture.

For a list of counter buses and their assigned channels and timebase sources, see Counter buses, channels, and timebase sources.

60.6.3.2 UC block diagram

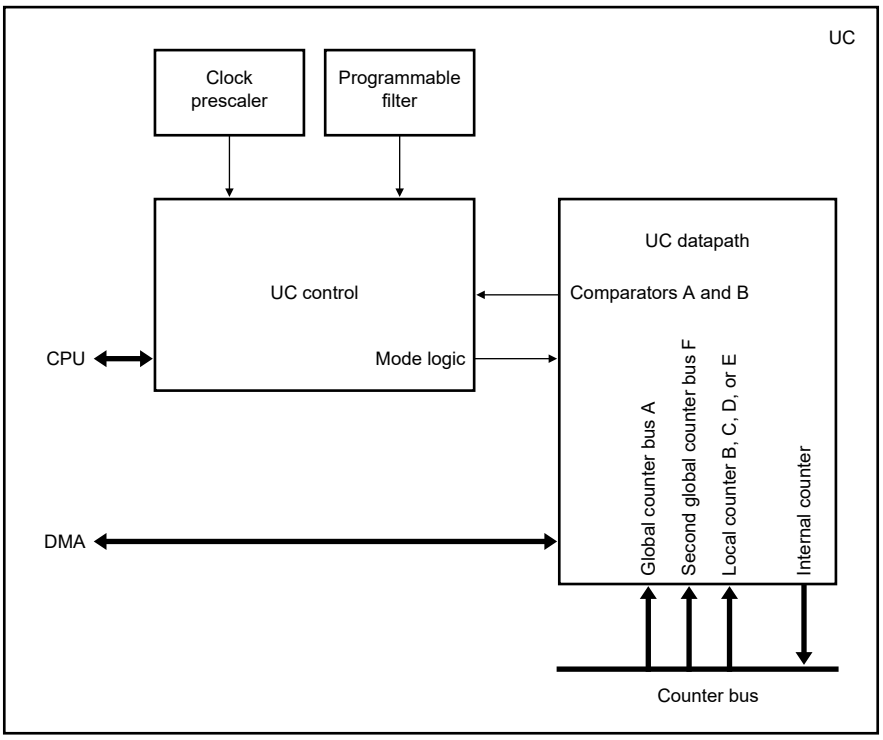


Figure 298. UC block diagram

60.6.3.3 Buffered modes

To provide smooth waveform generation even when A*n*[A] and B*n*[B] are changing, the following modes double buffer the A*n* and B*n* registers:

- Modulus Counter Buffered (MCB) mode
- Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode
- Output PWM Buffered (OPWMB) mode

These modes appear in separate sections because there are several basic differences in UC operation between them and their corresponding nonbuffered modes.

60.6.3.4 UC control and datapath

As illustrated in UC control and datapath diagram, the UC contains control and datapath subblocks.

The control subblock generates signals to control the multiplexers in the datapath subblock. eMIOS implements each UC mode in dedicated logic independent from the other modes. The UC modes share a set of registers allowing the storing of sequential events.

The datapath block contains the channel A and B registers, the internal timebase, and the comparators. Multiplexers receive inputs from the control subblock to configure the datapath for the specific channel modes. This configuration consists of selecting the input of comparators and data for the register inputs. The outputs of the A and B comparators connect to the control block.

60.6.3.5 UC control and datapath diagram

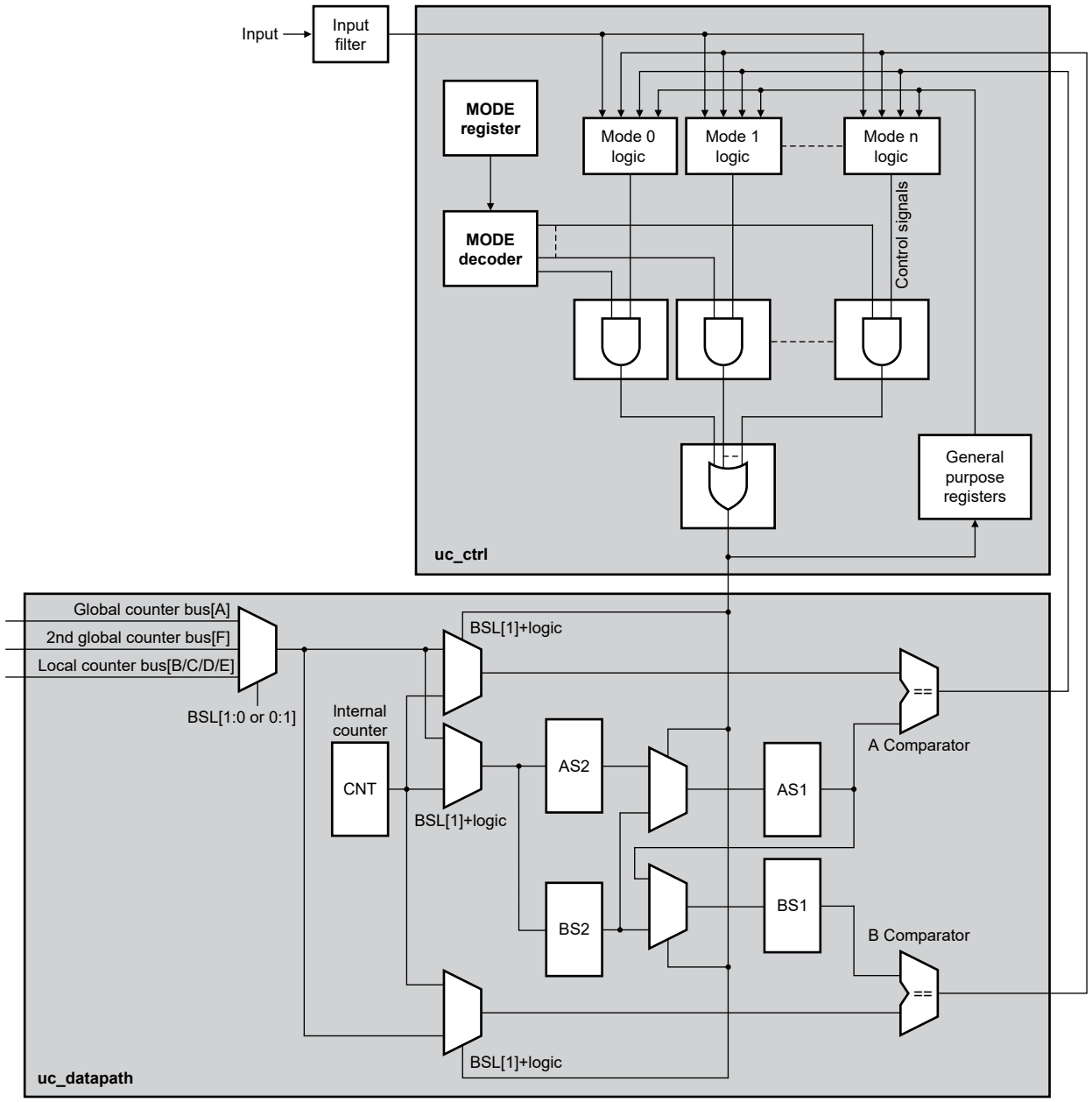


Figure 299. UC control and datapath diagram

60.6.3.6 UC modes

You can configure UCs to operate in the following modes:

- General-Purpose Input and Output (GPIO) mode
- Single Action Input Capture (SAIC) mode
- Single Action Output Capture (SAOC) mode
- Input Pulse Width Measurement (IPWM) mode
- Input Period Measurement (IPM) Mode

- Double Action Output Compare (DAOC) mode
- Pulse Edge Counting (PEC) mode
- Modulus Counter (MC) mode
- Modulus Counter Buffered (MCB) mode
- Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode
- Center Aligned Output PWM with Dead Time Insertion Buffered (OPWMCB) mode
- Output PWM Buffered (OPWMB) mode
- Output PWM with Trigger (OPWMT) mode

Each channel supports a specific subset of these modes. See the chip-configuration information to see which modes each channel supports.

60.6.3.7 General-Purpose Input and Output (GPIO) mode

60.6.3.7.1 Overview

This mode disables all UC input-capture and output-compare functions, and resets and disables the internal counter (C*n*[C]). All control fields remain accessible.

To change the UC from one operation mode to another, you must first change to GPIO mode and then change to the desired final mode. To prepare the UC for a new operation mode, you must configure AS1, BS1, AS2, and BS2 before exiting GPIO mode and entering the desired final mode (see AS1, AS2, BS1, and BS2 shadow registers). Specifically:

- Writing to A*n*[A] stores the same value in AS1 and AS2.
- Writing to B*n*[B] stores the same value in BS1 and BS2.
- Writing to ALTA*n*[ALTA] stores a value only in AS2.

As its name implies, GPIO mode supports both input and output submodes. See Differences for input and output submodes.

60.6.3.7.2 Differences for input and output submodes

Table 332. Differences for input and output submodes

Submode	To enter submode, write this value to C*n*[MODE]	Characteristics
Input	0	• You control the generation of input-capture flags with C*n*[EDPOL] and C*n*[EDSEL]. • You can determine the input pin status by reading S*n*[UCIN].
Output	1	• The UC performs as a single output port pin. • The value of C*n*[EDPOL] permanently drives the output flip-flop.

60.6.3.8 Single Action Input Capture (SAIC) mode

60.6.3.8.1 Overview

In this mode, when a triggering event occurs on the input pin, eMIOS:

- Captures the selected timebase into AS2
- Changes S*n*[FLAG] to 1 to indicate that an input capture has occurred

A*n*[A] returns the value of AS2. As soon as the UC enters SAIC mode after exiting GPIO mode, the channel is ready to capture events.

eMIOS captures the events as soon as they occur. Therefore, reading A*n*[A] always returns the value of the latest captured event. Subsequent capture events occur regardless of whether you read the previous event from A*n*[A]. Each new capture event sets the input capture flag (writes 1 to S*n*[FLAG]).

The input capture is triggered by an edge transition on the input pin as configured by C*n*[EDPOL] and C*n*[EDSEL].

SAIC with rising edge triggering and SAIC with both edges triggering show how to use the UC for input capture.

60.6.3.8.2 SAIC submodes and MODE field values

Table 333. SAIC submodes and MODE field values

Capture behavior	C*n*[MODE] (binary)
The UC only captures the timestamp.	000_0010
• The UC captures a timestamp. • AS2 bit 0 indicates the input signal level.	100_0010

60.6.3.8.3 Effect of C*n*[EDPOL] on edge capturing

Table 334. Effect of C*n*[EDPOL] on edge capturing

C*n*[EDPOL]	Input signal transition	Resulting value of AS2 bit 0
0	0→1	0
	1→0	1
1	0→1	1
	1→0	0

60.6.3.8.4 SAIC with rising edge triggering

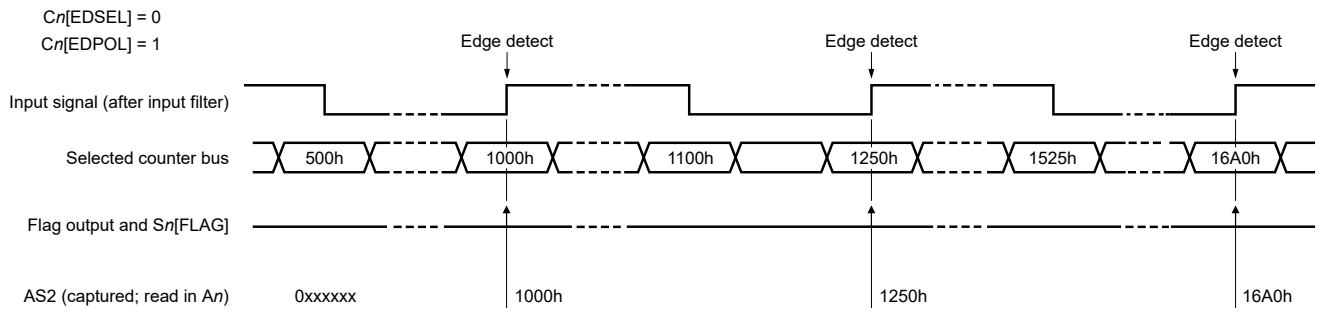


Figure 300. SAIC with rising edge triggering

60.6.3.8.5 SAIC with both edges triggering

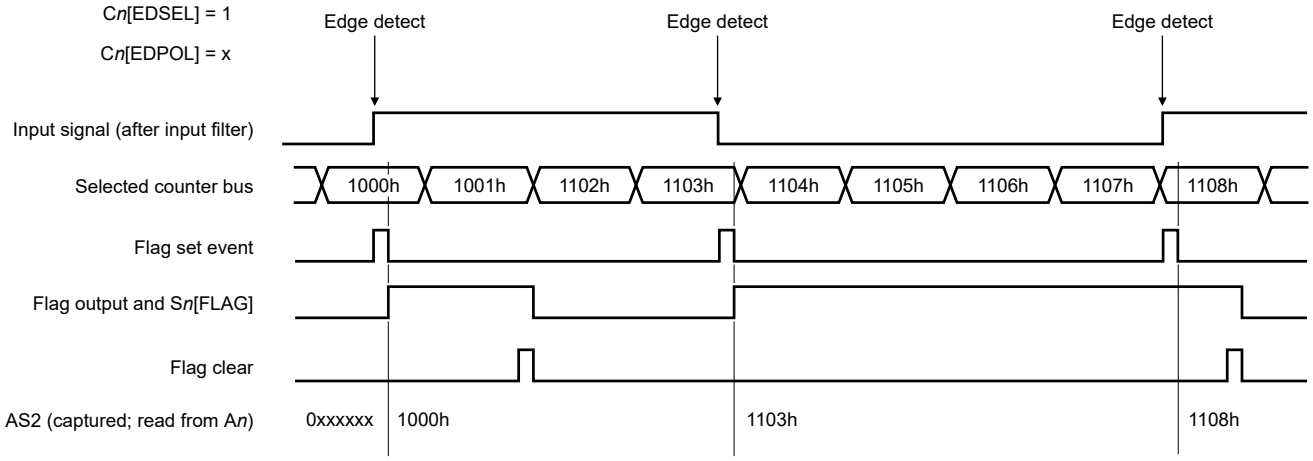


Figure 301. SAIC with both edges triggering

60.6.3.9 Single Action Output Capture (SAOC) mode

60.6.3.9.1 Overview

In this mode, eMIOS loads a match value into AS2 and then immediately transfers it to AS1 for comparison with the selected timebase. When an output-compare match occurs, eMIOS:

- Either toggles the output flip-flop or transfers C*n*[EDPOL] to the output flip-flop, as determined by C*n*[EDSEL].
- Sets the input capture flag (changes S*n*[FLAG] to 1).

Along with the match, S*n*[FLAG] becomes 1 to indicate that the output-compare match has occurred. Writing to A*n*[A] stores the value in AS2, and reading A*n*[A] returns the AS1 value.

The channel internal counter in SAOC mode is free-running. It starts counting as soon as the UC enters SAOC mode.

You can force an output-compare match by writing 1 to C*n*[FORCMA]. A forced output-compare match does not set the input capture flag (S*n*[FLAG]).

When the UC enters SAOC mode after exiting GPIO mode, the output flip-flop value becomes the complement of C*n*[EDPOL].

You select the internal or external counter bus with C*n*[BSL].

SAOC with C*n*[EDPOL] transferred to the output flip-flop shows how to perform a single output compare and transfer C*n*[EDPOL] to the output flip-flop.

SAOC toggling the output flip-flop shows how to perform a single output compare and toggle the output flip-flop at each match.

60.6.3.9.2 Preload the desired match value

eMIOS enables matches immediately after the UC enters SAOC mode. Therefore, you must write the desired match value to AS1 before the UC enters SAOC mode. You can update AS1 at any time. This modifies the match value reflected in the channel-generated output signal. Subsequent capture events occur regardless of whether you write to A*n*[A] again. Each new capture event sets the input capture flag (S*n*[FLAG]). See SAOC with flag behavior.

60.6.3.9.3 SAOC with C*n*[EDPOL] transferred to the output flip-flop

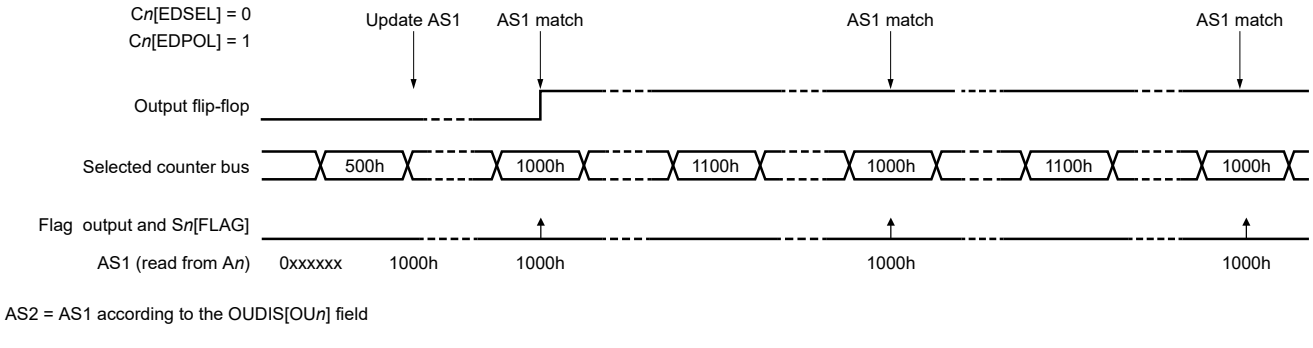


Figure 302. SAOC with C*n*[EDPOL] transferred to the output flip-flop

60.6.3.9.4 SAOC toggling the output flip-flop

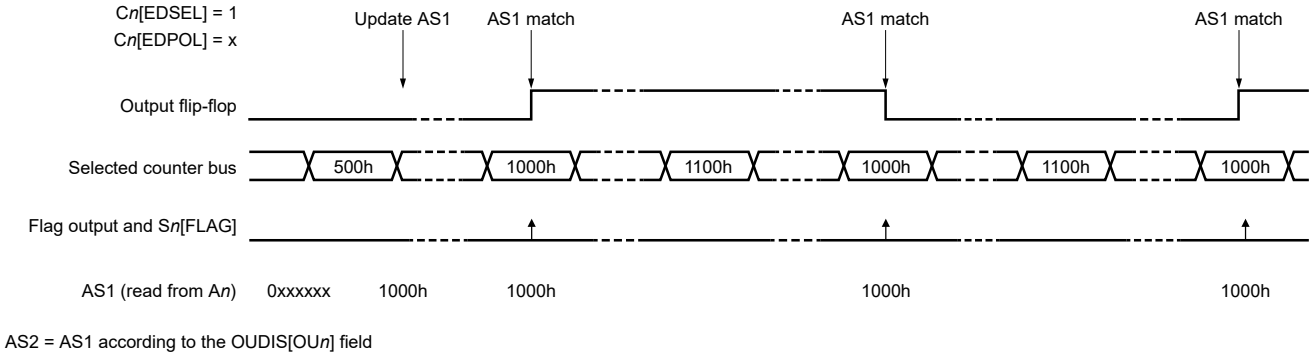


Figure 303. SAOC toggling the output flip-flop

60.6.3.9.5 SAOC with flag behavior

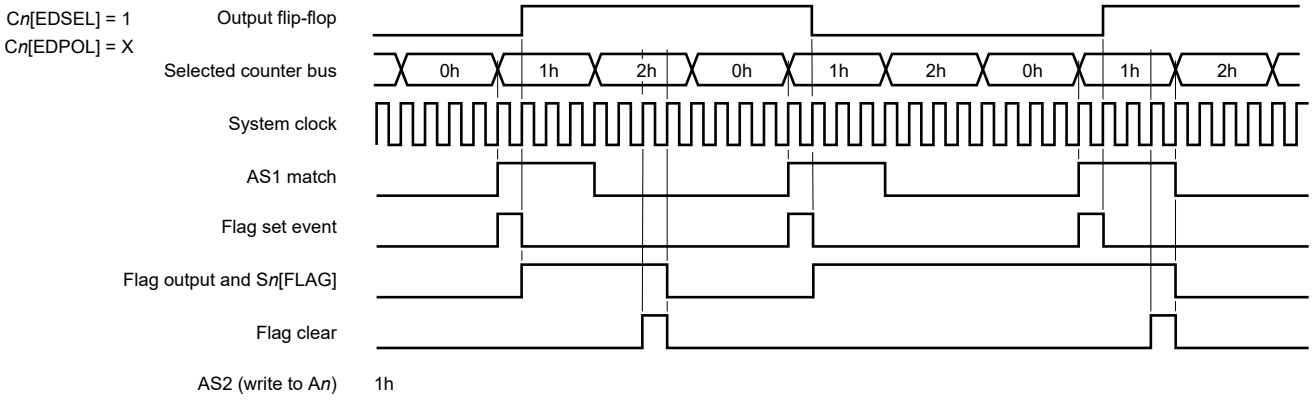


Figure 304. SAOC with flag behavior

60.6.3.10 Input Pulse Width Measurement (IPWM) mode

60.6.3.10.1 Overview

This mode allows you to measure the width of a positive or negative pulse. Select whether to measure from the rising or falling edge with Edge Polarity (Cn[EDPOL]).

The first leading edge:

- Triggers the first input capture, which latches the count value of the selected timebase into BS2 (see AS1, AS2, BS1, and BS2 shadow registers)
- Does not set the input-capture flag (Sn[FLAG])
- Enables AS2 capture

The next trailing edge:

- Latches the selected timebase into AS2
- Sets the input-capture flag (Sn[FLAG])
- Transfers BS2 to BS1 and AS1

eMIOS captures successive values on consecutive edges of opposite polarity. If you initiate subsequent input capture events while the input-capture flag (Sn[FLAG]) is set, eMIOS updates AS2, BS1, and AS1 with the latest captured values and keeps the flag set.

60.6.3.10.2 Coherency

Reading A*n*[A] updates BS1 with AS1. It also disables transfers from BS2 and BS1 until the next read of B*n*[B].

Reading Bn updates BS1 with AS1. It also re-enables transfers from BS2 to BS1, which takes effect at the next trailing edge capture.

Because transfers from BS2 to AS1 are never blocked, to guarantee data coherency you must read An, then read Bn.

If you do not require coherent data, you should read Bn before you read An, although doing so requires a second read of Bn to unblock BS1 updates.

60.6.3.10.3 Timing example: measure pulse width

This figure shows how to measure the input pulse width by subtracting BS1 from AS2.

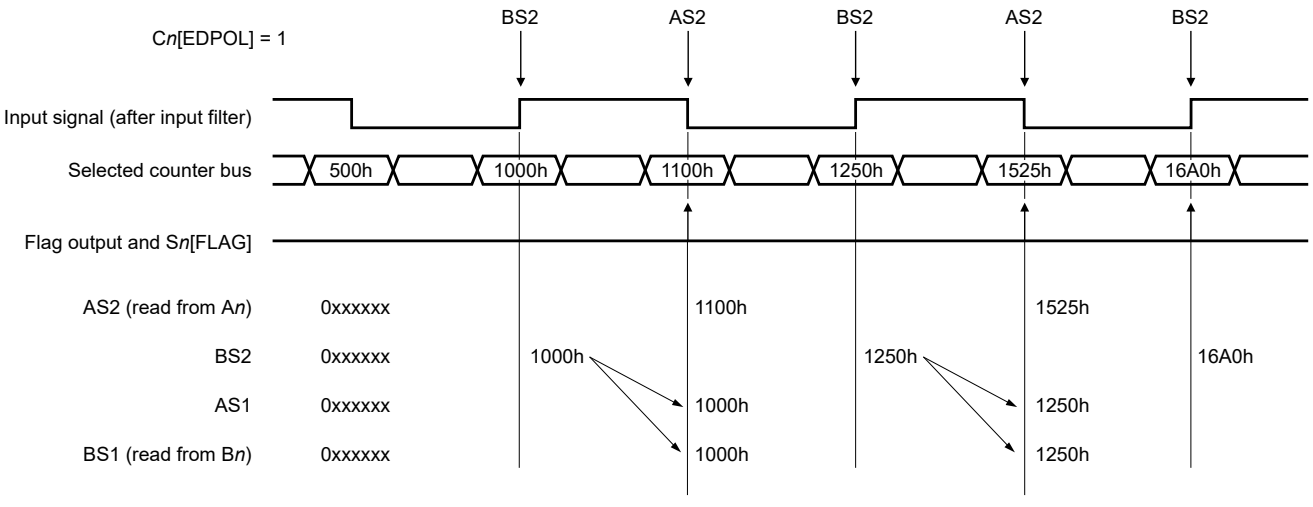


Figure 305. Timing example: measure pulse width

60.6.3.10.4 Timing example: AS1 and BS1 updates on channel reads

This example illustrates AS1 and BS1 updates when you read A*n*[A] and B*n*[B]. AS1 always has coherent data related to AS2 (see Coherency).

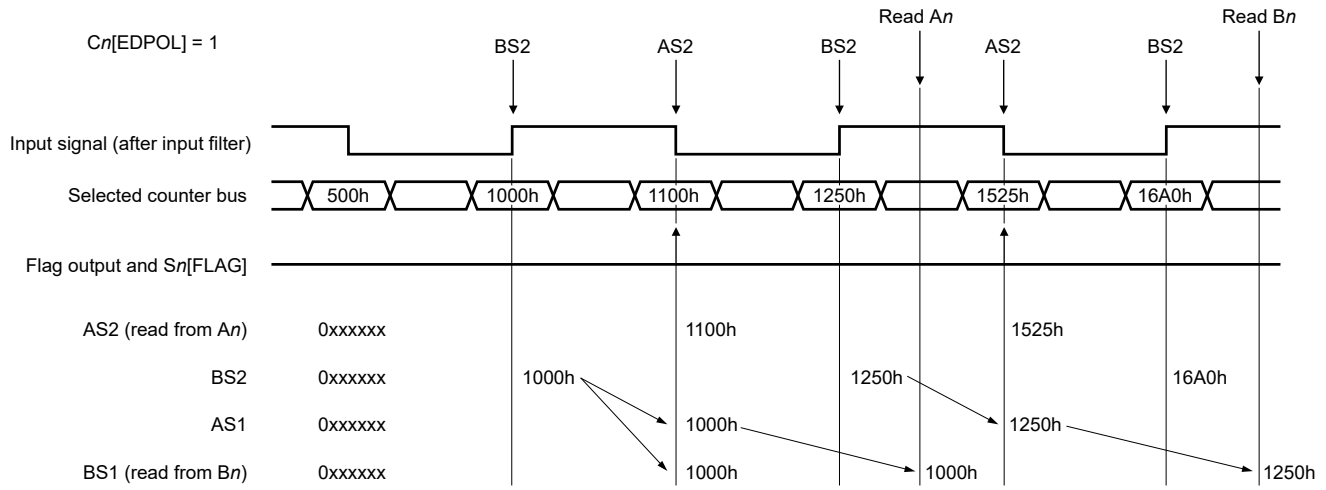


Figure 306. Timing example: BS1 and AS1 updates on channel reads

60.6.3.11 Input Period Measurement (IPM) Mode

60.6.3.11.1 Overview

This mode allows you to measure the period of an input signal. Select whether to measure from the rising or falling edges with Edge Polarity (C*n*[EDPOL]).

The first edge of selected polarity:

- Latches the selected timebase into AS2 and BS2 (see AS1, AS2, BS1, and BS2 shadow registers)
- Transfers BS2 to BS1
- Does not set the input-capture flag (S*n*[FLAG])

The second edge of the same polarity:

- Latches the counter bus value into AS2 and BS2
- Transfers BS2 to BS1 and AS1
- Sets the input-capture flag (S*n*[FLAG])

eMIOS repeats this process for each subsequent capture.

eMIOS captures successive values on two consecutive edges of the same polarity. The measured input signal must have a period of at least four system clock cycles to be properly captured by the synchronization logic at the channel input, even if the input filter is in bypass mode.

60.6.3.11.2 Coherency

Reading A*n*[A] updates BS1 with AS1, which provides coherent data in AS2 and BS1 (see Timing example: AS1 and BS1 updates on channel reads). It also disables transfers from BS2 and BS1 until the next read of B*n*[B].

Reading B*n*[B] updates BS1 with AS1. It also re-enables transfers from BS2 to BS1, which takes effect at the next edge capture.

60.6.3.11.3 Timing example: measure input period

This example illustrates how to measure the input period by subtracting BS1 from AS2.

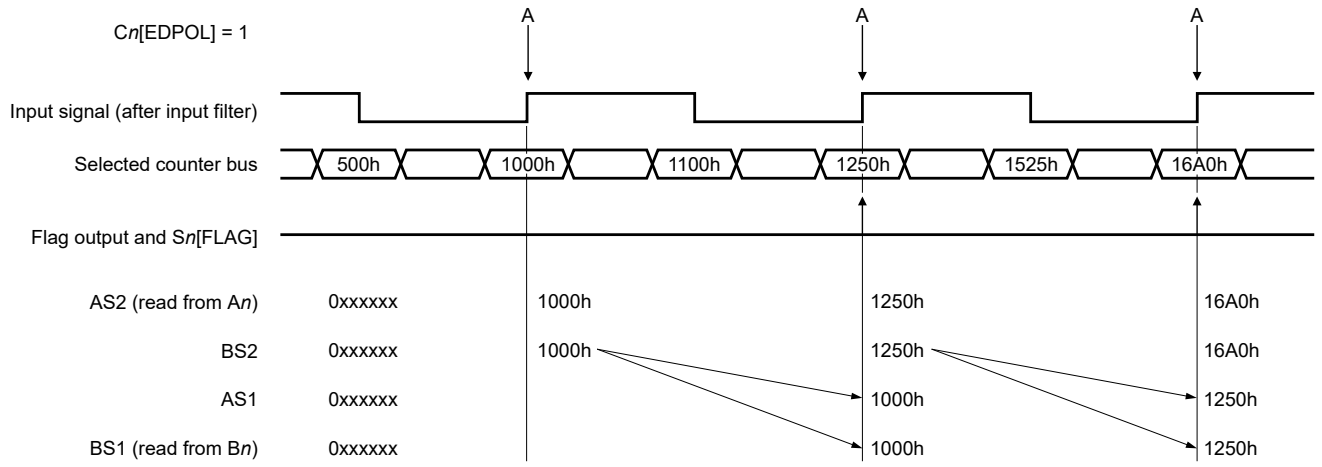


Figure 307. Timing example: measure input period

60.6.3.11.4 Timing example: AS1 and BS1 updates on channel reads

This diagram illustrates AS1 and BS1 updates when you read A*n*[A] and B*n*[B].

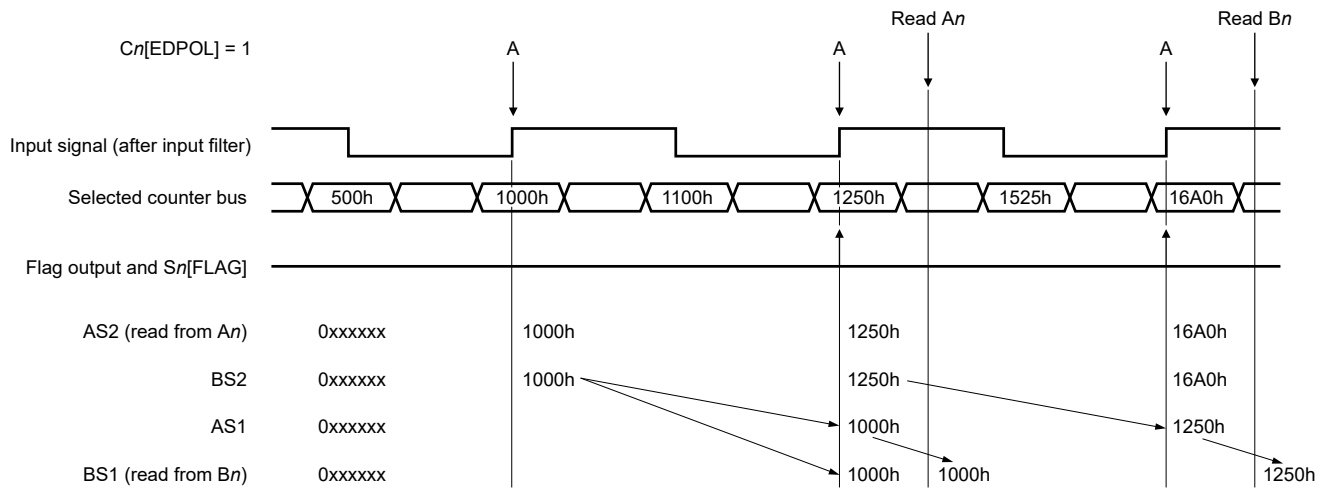


Figure 308. Timing example: AS1 and BS1 updates on channel reads

60.6.3.12 Double Action Output Compare (DAOC) mode

60.6.3.12.1 Overview

In this mode, matches on comparators A and B cause the UC to generate the leading and trailing edges of the variable pulse width output. See DAOC submodes and MODE field values. There is no restriction on the order in which A and B matches occur.

When the UC enters DAOC mode from GPIO mode, eMIOS disables both comparators and drives the complement of Edge Polarity (C*n*[EDPOL]) onto the output flip-flop.

60.6.3.12.2 DAOC submodes and MODE field values

Table 335. DAOC submodes and MODE field values

When set flag event occurs	C*n*[MODE] (binary)
Only on B matches	000_0110
On A or B matches	000_0111

60.6.3.12.3 Data transfer and compare

If you enable output (write 0 to OUDIS[OU*n*]), on the next clock cycle the UC transfers AS2 and BS2 to AS1 and BS1 respectively (see Timing example: leading dead time insertion). If you disable output (write 1 to OUDIS[OU*n*]), no transfer occurs.

eMIOS enables and disables the A and B comparators independently. It enables comparator A only after the transfer to AS1 completes, and disables comparator A on the next A match. Similarly, eMIOS enables comparator B only after the transfer to BS1 completes, and disables comparator B on the next B match.

When a match occurs on comparator A, the output flip-flop acquires the value of Edge Polarity (C*n*[EDPOL]). When a match occurs on comparator B, the output flip-flop acquires the complement of Edge Polarity (C*n*[EDPOL]).

60.6.3.12.4 Differences between flag on A match and flag on A and B matches

In the flag on B match submode (see DAOC submodes and MODE field values), the UC sets the input-capture flag (S*n*[FLAG]) only for matches on the B comparator. In the flag on both A and B matches submode, it sets the flag for matches on either comparator.

If subsequent enabled output compares occur on AS1 or BS1, the UC continues generating pulses regardless of the flag state.

If you load both AS1 and BS1 with the same value, the B match takes precedence and the UC drives the complement of Edge Polarity (C*n*[EDPOL]) onto the output flip-flop.

60.6.3.12.5 Forced operation by Cn[FORCMA] and Cn[FORCMB]

Force Match A (C*n*[FORCMA]) and Force Match B (C*n*[FORCMB]) allow you to force the output flip-flop to the level corresponding to the comparator, Edge Polarity (C*n*[EDPOL]) for an A match, or the complement of Edge Polarity for a B match. The force-match operations do not set the input-capture flag (S*n*[FLAG]).

60.6.3.12.6 Timing example: flag on both matches

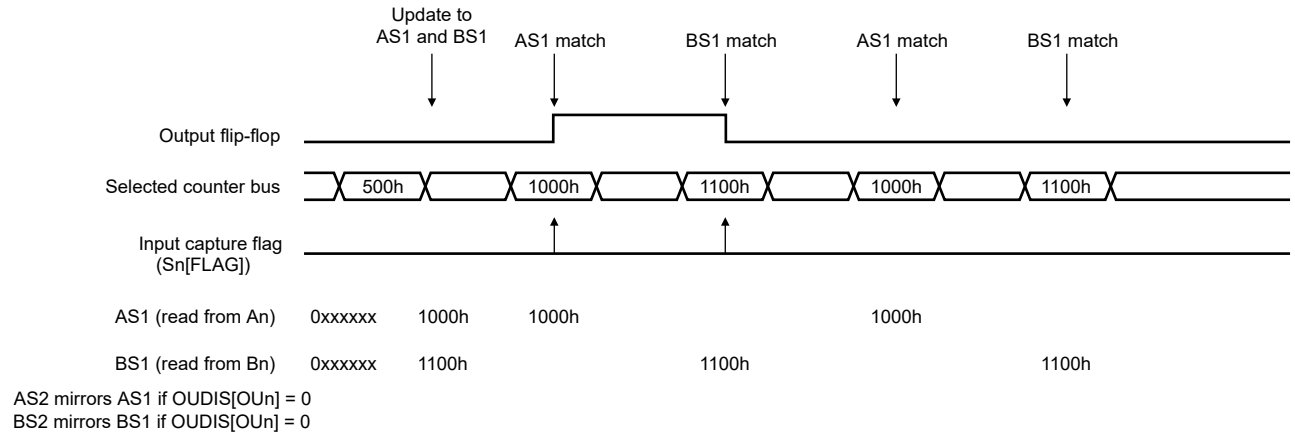


Figure 309. Timing example: flag on both matches

60.6.3.12.7 Timing example: flag on second match

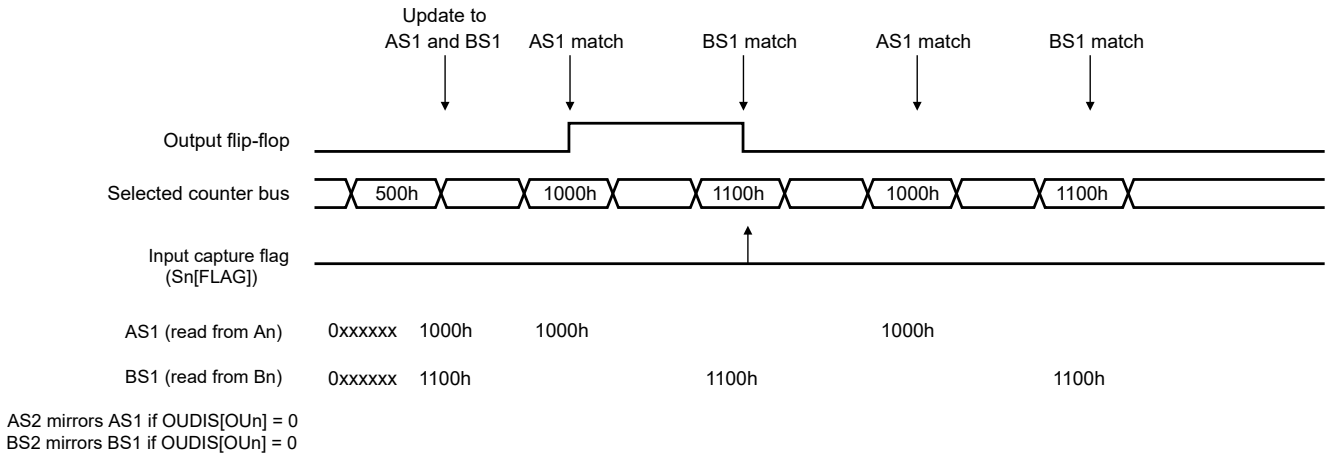


Figure 310. Timing example: flag on second match

60.6.3.12.8 Timing example: transfer disabling

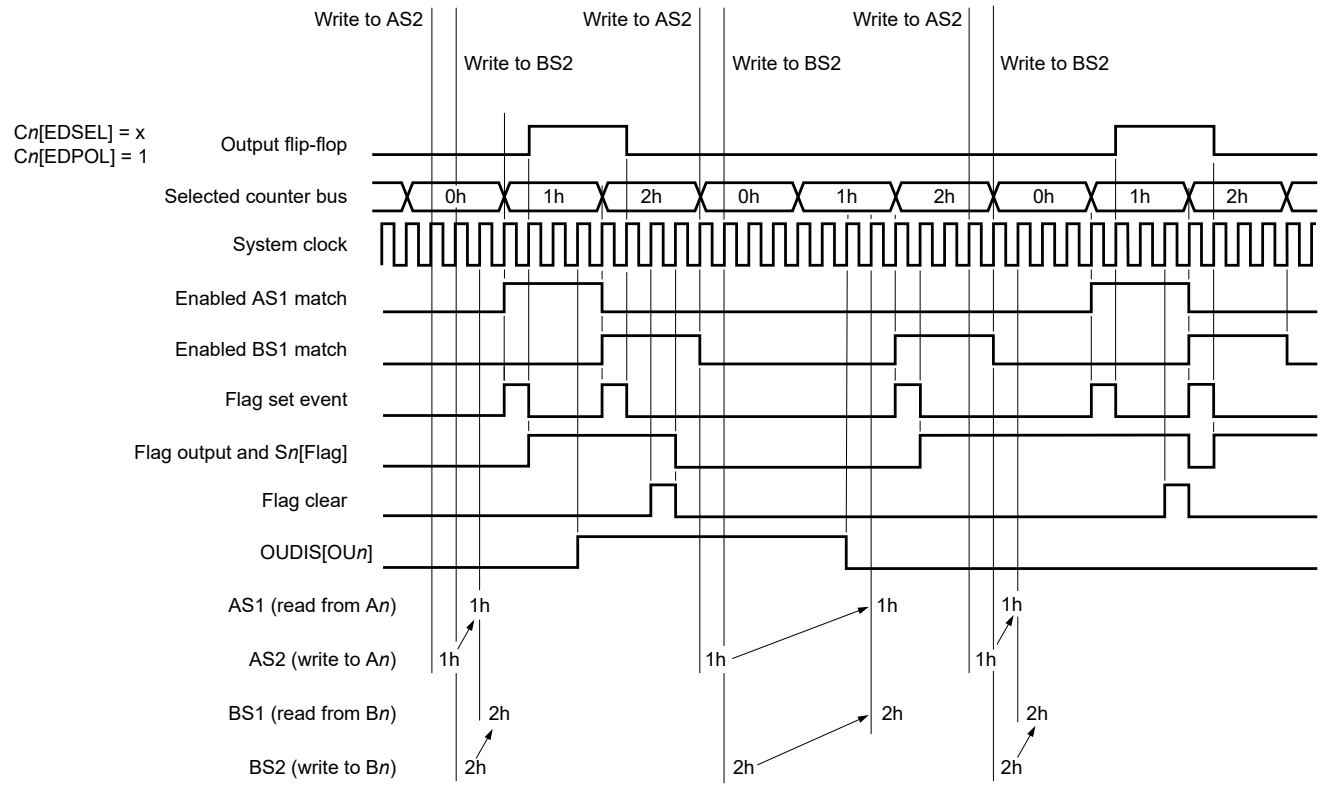


Figure 311. Timing example: transfer disabling

60.6.3.13 Pulse Edge Counting (PEC) mode

60.6.3.13.1 Overview

This mode counts the number of pulses or edges detected on the input for a desired time window. C*n*[EDSEL] and C*n*[EDPOL] determine which edges the UC counts.

Specify the window start time in AS1 and the end time in BS1. After you write to AS1, when the selected timebase matches comparator A, eMIOS resets the internal counter and starts counting input events. When the timebase matches comparator B, eMIOS:

- Disables the internal counter.
- Transfers the counter's content to AS2.
- Sets the input-capture flag (changes S*n*[FLAG] to 1).

You obtain the number of detected pulses by reading C*n*[C] or AS2.

Timing example continuous and Timing example single-shot show how to use the UC in continuous or single-shot operation.

60.6.3.13.2 PEC submodes and MODE field values

Table 336. PEC submodes and MODE field values

Operation	C*n*[MODE] (binary)
Continuous; the next match between comparator A and the selected timebase resets the internal counter and enables counting again. To guarantee coherent measurements when you read C*n*[C] after S*n*[FLAG] becomes 1, you must check if the timebase value is out of the time interval defined by AS1 and BS1. Alternately, AS2 (available in ALTA*n*[ALTA]) always holds the latest available measurement, providing coherent data at any time after the first set-flag event.	000_1010
Single-shot; The next match between comparator A and the selected timebase has no effect until you write to A*n*[A]. eMIOS also transfers the C*n*[C] value to AS2 when a match in the B comparator occurs.	000_1011

60.6.3.13.3 Timing example continuous

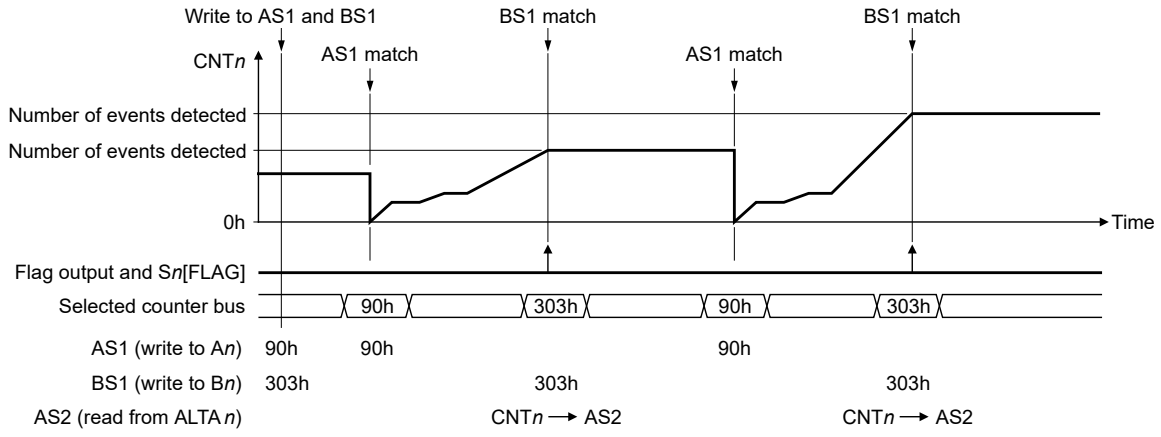


Figure 312. Timing example continuous

60.6.3.13.4 Timing example single-shot

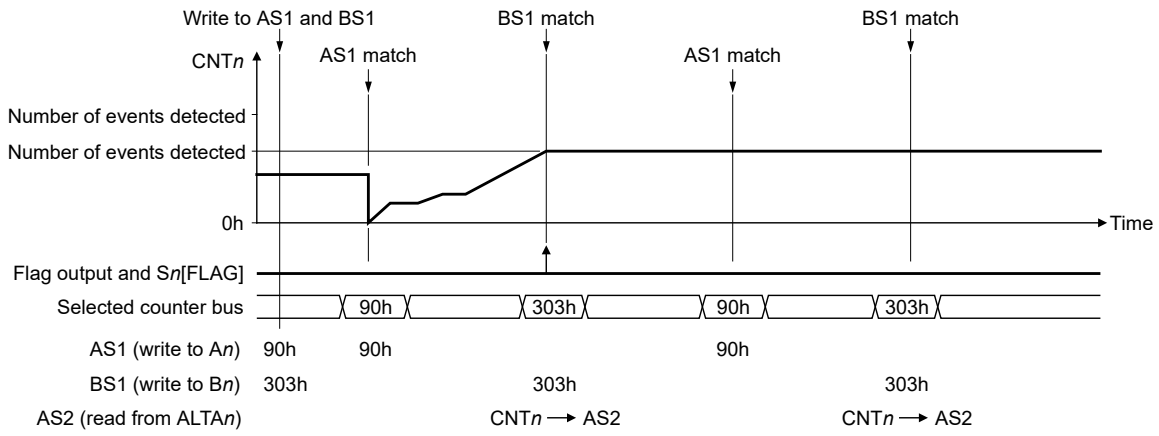


Figure 313. Timing example single-shot

60.6.3.14 Modulus Counter (MC) mode

60.6.3.14.1 Overview

In this mode, the UC produces a timebase for a counter bus or as a general purpose timer. The internal counter (C*n*[C]) counts up from the current value until it matches AS1.

Selecting external clock source (see MC submodes and MODE field values) causes the UC to uses the input signal pin as the source. You select the triggering polarity edge using Edge Polarity (C*n*[EDPOL]) and Edge Select (C*n*[EDSEL]).

You must write only non-zero values to A*n*[A]. Writing to A*n*[A] or the internal counter may cause a match to be missed and the counter to roll over and resume operation in the next cycle. The channel writes 0 to BS1, which is not accessible by the MCU. You can read and write BS2, but it is not used in this mode.

60.6.3.14.2 MC submodes and MODE field values

Table 337. MC submodes and MODE field values

Submode or function	C*n*[MODE] (binary)
Internal clock source	001_0pp0[1]
External clock source	001_0pp1[1]
Internal counter reset on match start	001_0p0p[1]
Internal counter reset on match end	001_0p1p[1]
Up Count	001_00pp[1]
Up Count-Down Count	001_01pp[1]

1. *p* = Adjust parameters for submodes and options. See Unified channels (UC).

60.6.3.14.3 Up Count submode

Table 338. Up Count submode operation

Counter reset timing[1]	C*n*[MODE]		
	Bit 1	External clock (Bit 0 = 1)	Internal clock (Bit 0 = 0)
Reset on match start	0	When the AS1 match occurs, the UC: • Writes 0 to the internal counter. • Sets the input-capture flag (S*n*[FLAG]). • Increments the internal counter at the next input event, resulting in a short zero count.	When the AS1 match occurs, the UC: • Writes 0 to the internal counter. • Sets the input-capture flag (S*n*[FLAG]). • Maintains the internal counter at 0 on the next prescaler tick after the match. • Resumes counting on the next prescaler tick.
		See Timebase with the fastest prescaler ratio and Timebase generation with clear on match start and internal clock.	
Clear on match end	1	When the next input event after the AS1 match occurs, the UC: • Writes 0 to the internal counter. • Sets the input-capture flag (S*n*[FLAG]).	When the next internal counter tick after the AS1 match occurs, the UC: • Writes 0 to the internal counter. • Sets the input-capture flag (S*n*[FLAG]).
		See Timebase with the fastest prescaler ratio and Timebase generation with clear on match end.	

1. If you select internal clock source and a prescaler value of 1 for the internal counter, the behavior is the same for both submodes.

See Timing example: Up Count submode.

60.6.3.14.4 Timing example: Up Count submode

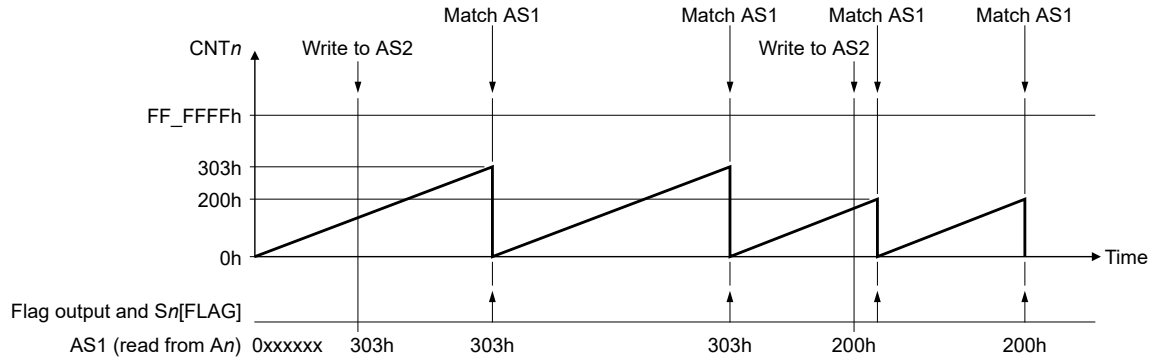


Figure 314. Timing example: Up Count submode

60.6.3.14.5 Up Count-Down Count submode operation

In Up Count-Down Count submode:

- When the internal counter matches AS1, the counter begins decrementing and the UC sets the input-capture flag (S*n*[FLAG]).
- When the internal counter matches BS1, the counter begins incrementing and, if in clear on match end operation (C*n*[1] = 1), the UC sets the input-capture flag (S*n*[FLAG]).

See Timing example: Up Count-Down Count submode.

60.6.3.14.6 Timing example: Up Count-Down Count submode

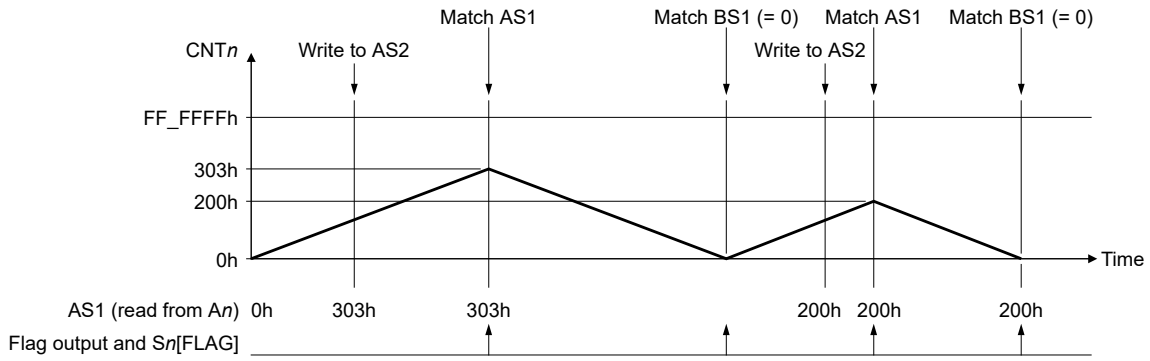


Figure 315. Timing example: Up Count-Down Count submode

60.6.3.14.7 Up Count with delayed reload operation

When the internal counter matches AS1 in the Up Count submode or BS1 in the Up Count-Down Count submode, the UC asserts the counter bus reload signal. You can read the counter bus reload signal in the following modes:

- Output PWM Buffered (OPWMB) mode
- Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode
- Center Aligned Output PWM with Dead Time Insertion Buffered (OPWMCB) mode

You can use the Reload Signal Output Delay Interval (C2*n*[UCRELDEL_INT]) to cause the UC to assert the reload signal only after a specified number of match events (from 2 to 32).

See Timing example: Up Count mode with delayed reload.

60.6.3.14.8 Timing example: Up Count mode with delayed reload

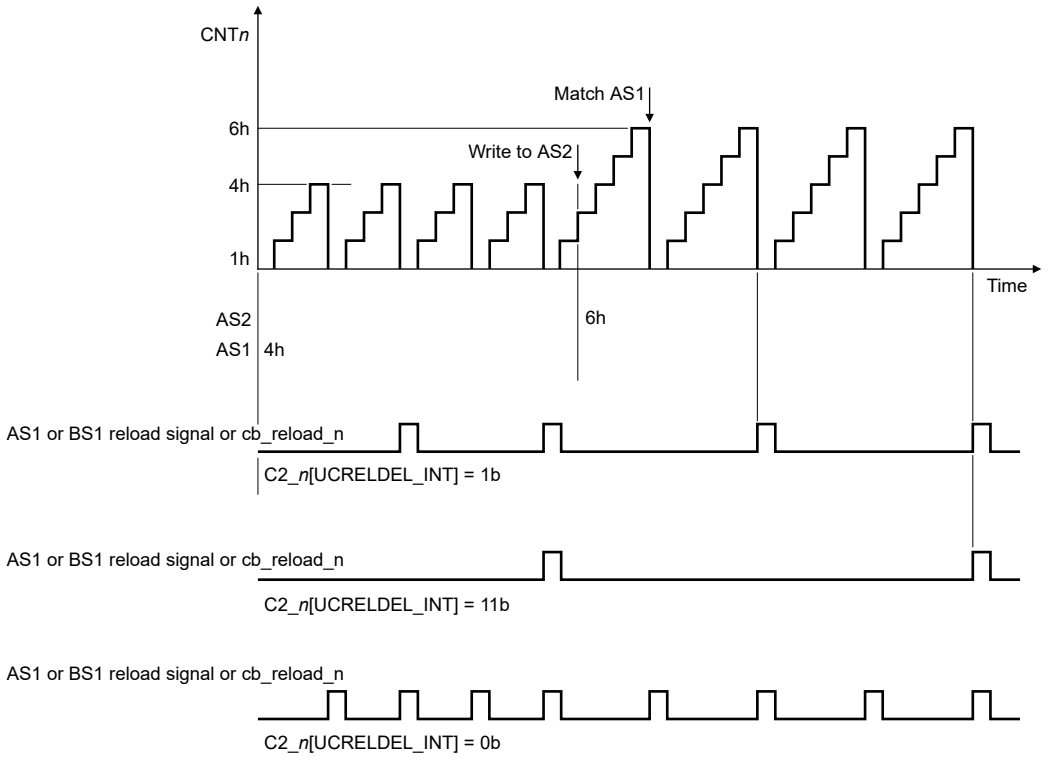


Figure 316. Timing example: Up Count mode with delayed reload

60.6.3.15 Modulus Counter Buffered (MCB) mode

60.6.3.15.1 Overview

In this mode, the UC generates a timebase that can be shared with other channels through the internal counter buses. AS1 is double-buffered, so you can change AS2 at any time with smooth transitions. The UC updates AS1 at the counter period boundary, which is when the internal counter (CNT*n*[C]) transitions to 1h.

Selecting external clock (see MCB submodes and MODE field values) causes the UC to use the input signal as the source. You select the triggering edge with Edge Polarity (C*n*[EDPOL]) and Edge Select (C*n*[EDSEL]).

You must ensure the internal counter is in the range from 1h to the AS1 value; otherwise, the A*n*[A] match does not occur and this causes the internal counter to wrap at the maximum counter value (FFFFh for a 16-bit counter). After a counter wrap occurs, the UC resets the counter to 1h and resumes normal operation.

NOTE

Do not write 1h to the internal counter when the UC is in Freeze state. Doing so can cause the match to be missed and the counter to overflow.

60.6.3.15.2 Changing to MCB mode

To avoid causing an unexpected interrupt or DMA request, you must use the following procedure to change to MCB mode:

1. Put the UC in GPIO mode (write 0b or 1b to C*n*[MODE]).
2. Disable the UC input capture flag (write 0 to C*n*[FEN]).
3. Write the correct values for MCB mode to A*n* and B*n*.

4. Put the UC in MCB mode (write the appropriate value to C*n*[MODE], as described in MCB submodes and MODE field values).
5. Clear the UC input capture flag (write 1 to S*n*[FLAG]).
6. Enable the UC input capture flag (write 1 to C*n*[FEN]).

60.6.3.15.3 MCB submodes and MODE field values

Table 339. MCB submodes and MODE field values

Submode or function	C*n*[MODE] (binary)
Internal clock source	101_0pp0b[1]
External clock source	101_0pp1b[1]
Up Count	101_000p[1]
Reserved	101_001p[1]
Up Count-Down Count	101_01pp[1]
Flag set only on match start	101_010p[1]
Flags also set at counter period boundary	101_011p[1]

1. *p* = Adjust parameters for submodes and options. See Unified channels (UC).

60.6.3.15.4 Up Count submode

In the Up Count submode:

- The internal counter (CNT*n*[C]) starts counting up (incrementing) from its current value.
- When the internal counter matches AS1 and a clock tick occurs (either prescaled clock or input pin event), the UC resets the internal counter to 1h.
- The UC sets the input-capture flag (S*n*[FLAG]) one system clock cycle after the match occurs.

You must write only values greater than 1h to AS1.

The boundary between counter period *n* and period *n*+1 occurs when the UC changes the internal counter from the AS1 value in period *n* to 1h in period *n*+1. AS1 defines the counter period.

The UC asserts the AS1 load signal (which has a duration of one system clock cycle) at the last system clock cycle of a counter period. This causes the UC to update AS1 with AS2 at the period boundary, at the same time that it resets the counter to 1h. Therefore, any value you write to AS2 during period *n* updates AS1 at the next period boundary and the UC uses it in period *n*+1. See Timing example: Up Count submode.

You can use Output Update Disable (OUDIS) to delay the update of AS1 for synchronization.

The UC sets the input-capture flag (S*n*[FLAG]) at the period boundary. See Timing example: Up Count submode flag assertion.

60.6.3.15.5 Timing example: Up Count submode

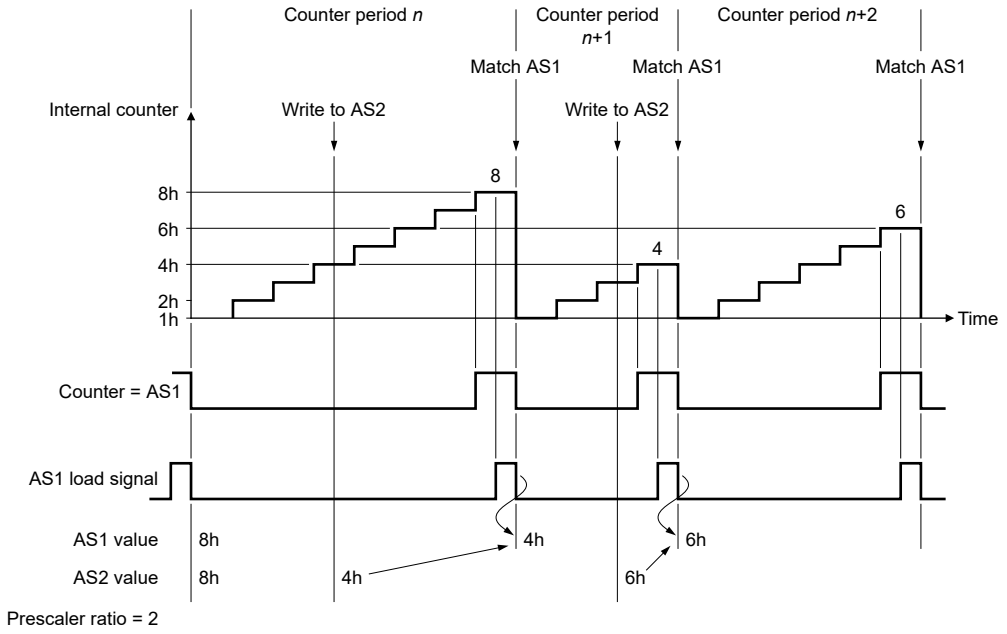


Figure 317. Timing example: Up Count submode

60.6.3.15.6 Timing example: Up Count submode flag assertion

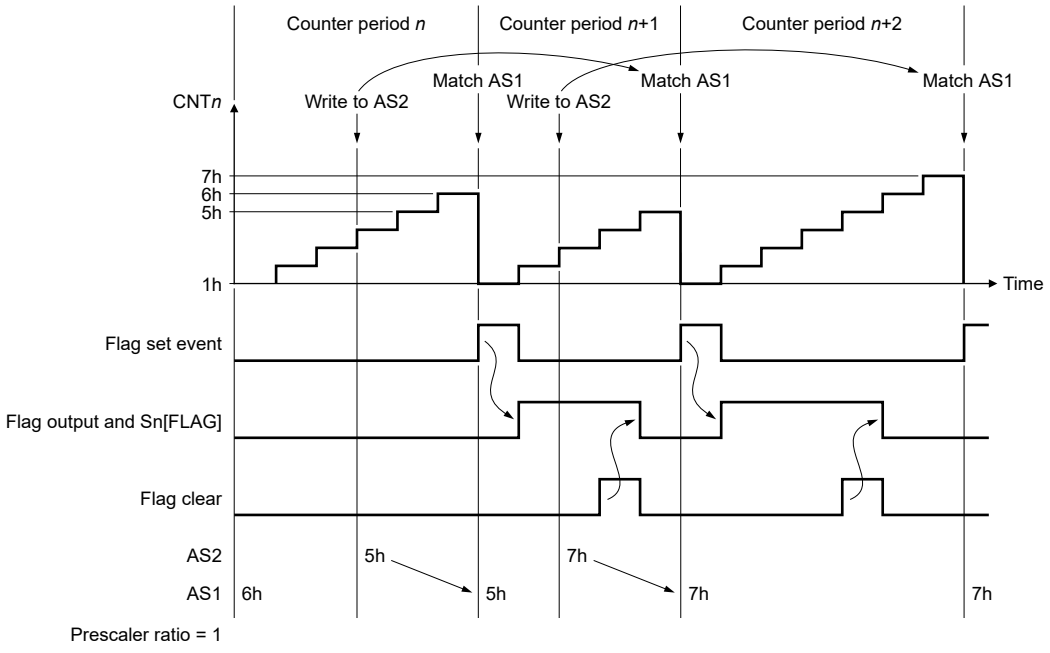


Figure 318. Timing example: Up Count submode flag assertion

60.6.3.15.7 Up Count-Down Count submode

In Up Count-Down Count submode:

- The counter period is: (2 x AS1) – 2. The counter changes direction at AS1 match and counts down (decrements) until it reaches 1h, then counts up (increments) again.
- BS1 generates a match to start the internal counter incrementing, and BS1 cannot be changed in this submode.

The UC updates AS1 at the counter period boundary. If you write AS2 (write A*n*[A]) during period *n*, the UC uses this new value in period *n*+1 for AS1 match. You can disable updates of AS1 (OUDIS[OU*n*]). See Timing example: Up Count-Down Count submode.

The UC sets the input-capture flag (S*n*[FLAG]) at AS1 match start, and may also do so at the period boundary. See Overview and Timing example: Up Count-Down Count submode flag assertion.

60.6.3.15.8 Timing example: Up Count-Down Count submode

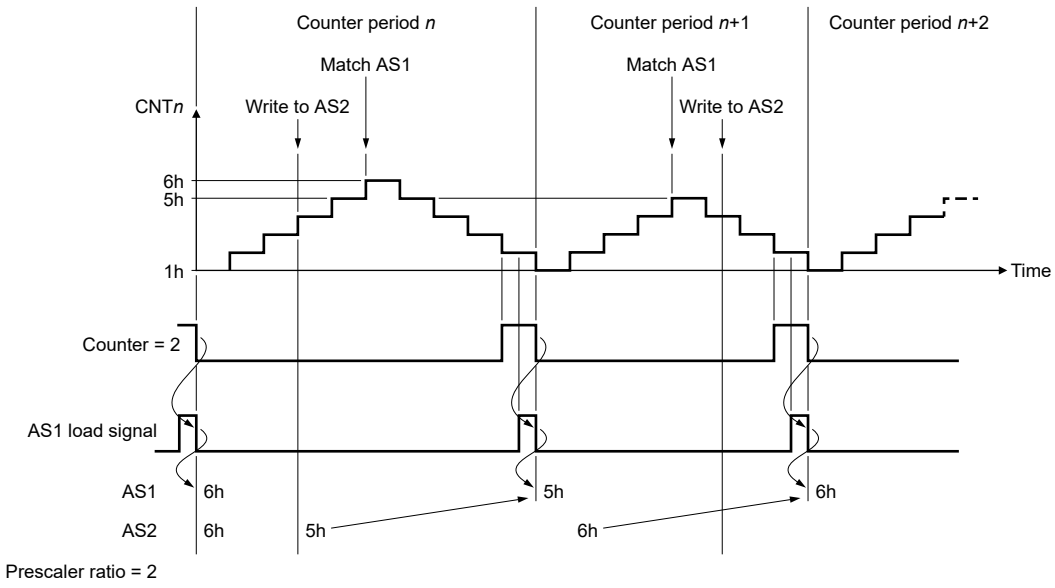


Figure 319. Timing example: Up Count-Down Count submode

60.6.3.15.9 Timing example: Up Count-Down Count submode flag assertion

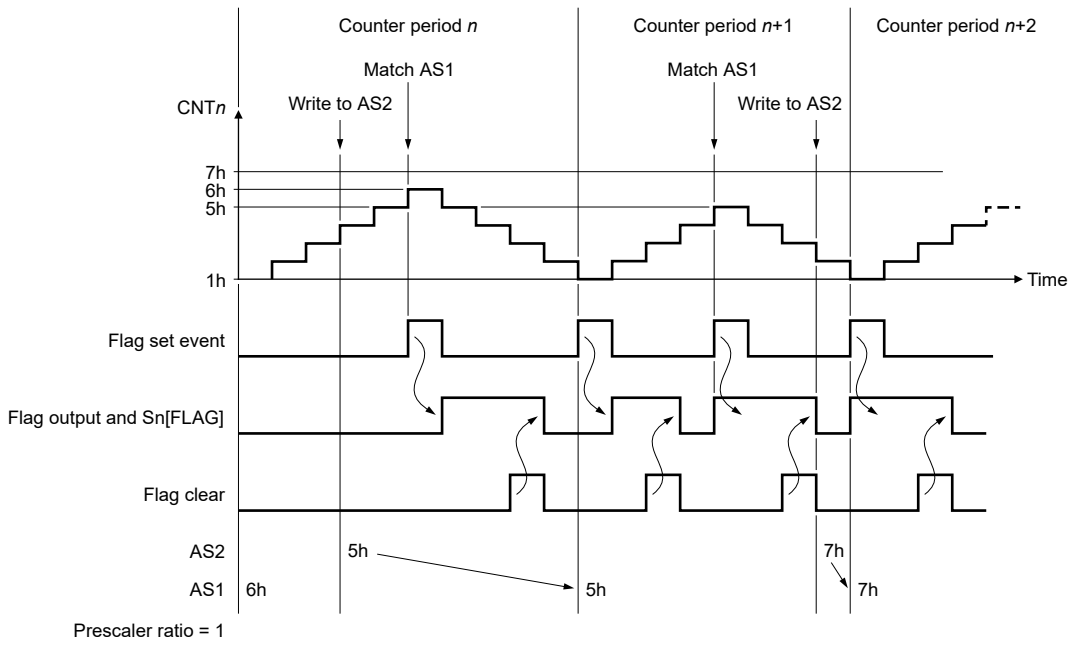


Figure 320. Timing example: Up Count-Down Count submode flag assertion

60.6.3.15.10 Up Count mode with delayed reload operation

When the internal counter matches AS1 in the Up Count submode or 1h in the Up Count-Down Count submode (see MCB submodes and MODE field values), the UC asserts the counter bus reload signal. You can read the counter bus reload signal in the following modes:

- Output PWM Buffered (OPWMB) mode
- Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode
- Center Aligned Output PWM with Dead Time Insertion Buffered (OPWMCB) mode

You can use the Reload Signal Output Delay Interval (C2[UCRELDEL_INT]) to cause the UC to assert the reload signal only after a specified number of match events (from 2 to 32).

See Timing example: Up Count mode with delayed reload.

60.6.3.15.11 Timing example: Up Count mode with delayed reload

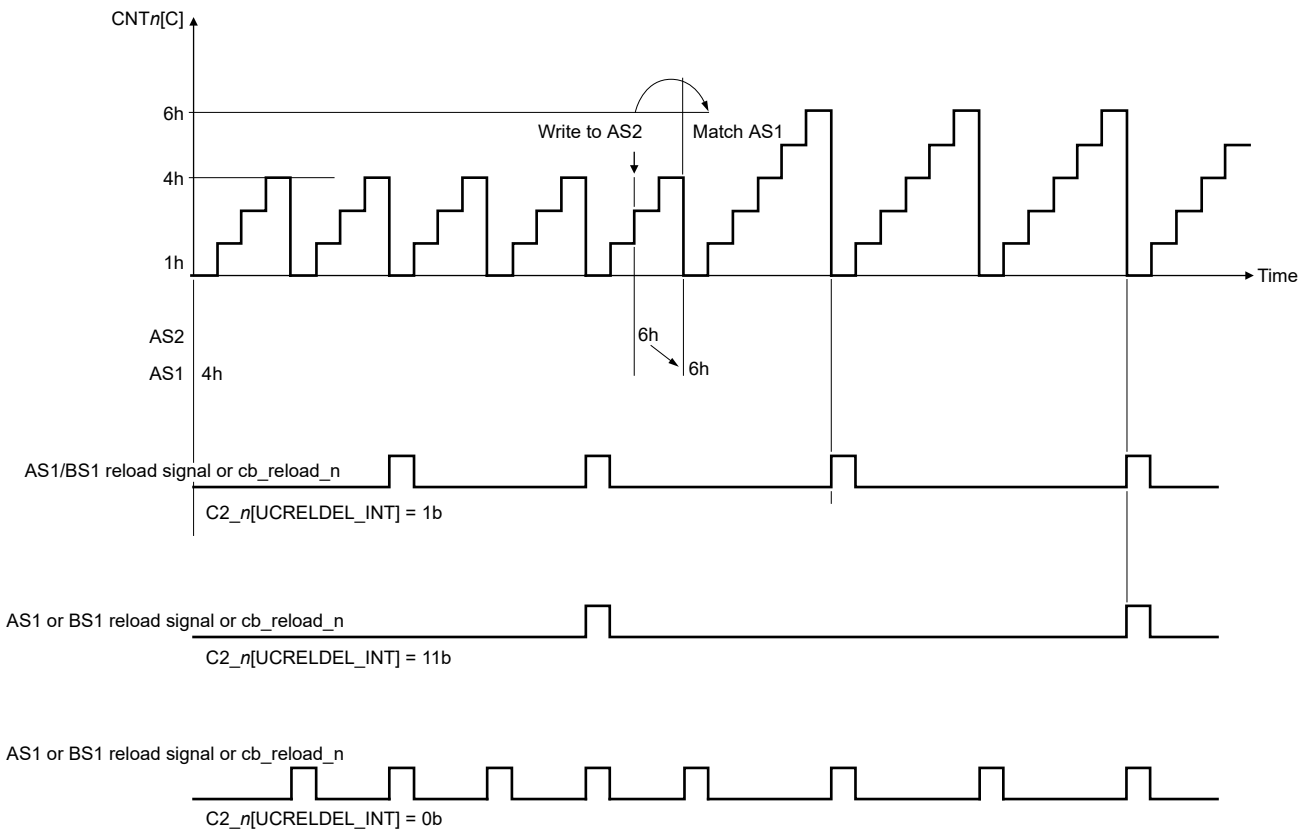


Figure 321. Timing example: Up Count mode with delayed reload

60.6.3.16 Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode

60.6.3.16.1 Overview

In this mode, the UC produces an output signal that has:

- A duty cycle determined by AS1.
- A period of BS1.

The double-buffered AS1 and BS1 registers produce smooth duty-cycle and period transitions when you change AS1 and BS1 during runtime.

See Table 346 for the values that you must write to C*n*[MODE] to enter this mode.

To provide smooth and consistent channel operation:

- AS1 and BS1 are both double-buffered to allow smooth signal generation when changing the register values.
- There is a delay from the AS1 match to the output pin transition (see AS1 and BS1 match timing).
- The internal counter ranges from 1h to BS1. When you write to BS1, you must write only values greater than 1h. Writing 0h or 1h leads to unpredictable results.

When a match on comparator A occurs, the UC drives C*n*[EDPOL] onto the output flip-flop. When a match on comparator B occurs, the UC drives the complement of C*n*[EDPOL] onto the output flip-flop. A BS1 match also causes the internal counter to transition to 1h, thus restarting the counter cycle.

The UC automatically selects the internal channel counter as the timebase when you select this mode. The mode supports duty cycles ranging from 0% to 100%.

60.6.3.16.2 OPWFMB submodes and MODE field values

Table 340. OPWFMB submodes and MODE field values

Match behavior	C*n*[MODE] (binary)
When BS1 matches the selected counter	101_1000
When AS1 or BS1 match the selected counter	101_1010

60.6.3.16.3 Avoid counter wrapping

When the UC enters OPWFMB mode, the output flip-flop acquires the value of C*n*[EDPOL]. If the UC transitions from GPIO to OPWFMB mode when the internal counter value is not within the range 1h–BS1, then the B match never occurs. Because the match never occurs, the internal counter wraps at the maximum counter value—FFFFh for a 16-bit counter. When the counter wrap occurs, the counter returns to 1h and normal OPWFMB operation resumes. Therefore, to avoid this counter wrapping, ensure that the counter value is within the range 1h–BS1 before the UC enters OPWFMB mode.

60.6.3.16.4 AS1 and BS1 match timing

As illustrated in Timing example: AS1 and BS1 match, the output pin transition occurs when the AS1 or BS1 match signal deasserts, as indicated by the AS1 match negative-edge detection signal. If you write 4h to AS1, the output pin transitions 4 counter increments plus one system clock cycle after the counter period starts.

60.6.3.16.5 Timing example: AS1 and BS1 match

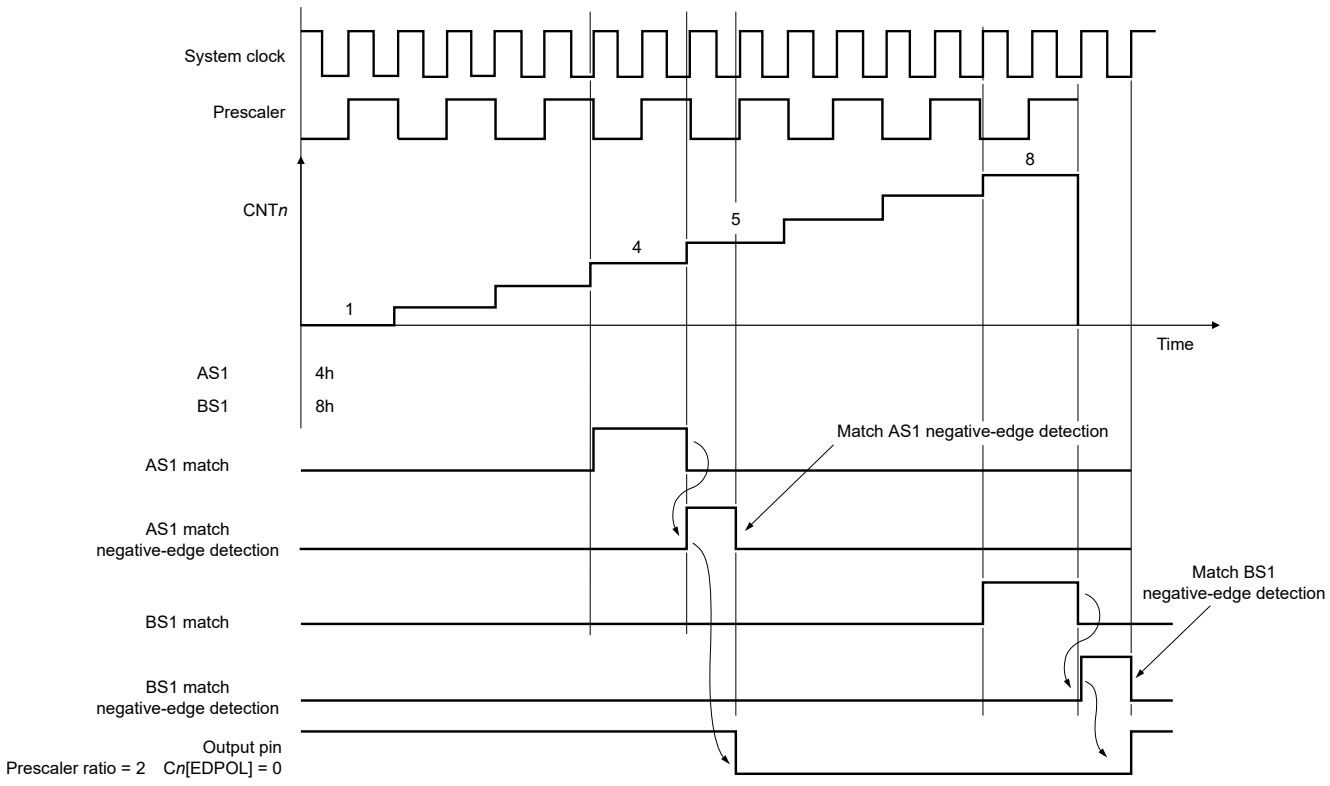


Figure 322. Timing example: AS1 and BS1 match

60.6.3.16.6 0% duty cycle

Timing example: AS1 = 0 illustrates the generated output signal when AS1 = 0, which results in a 0% duty cycle.

Because the internal counter never reaches zero in this situation, the UC infers a match as if AS1 = 1h. However, it is the positive edge of the match signal that triggers the output pin transition instead of the negative edge, which is what happens when AS1 = 1h. The AS1 positive-edge match signal from period *n*+1 occurs at the same time as the BS1 negative-edge match signal from period *n*. This timing allows the AS1 positive-edge match to mask the BS1 negative-edge match when they occur at the same time. As a result, no transition occurs on the output flip-flop and the UC generates a 0% duty cycle.

60.6.3.16.7 Timing example: AS1 = 0

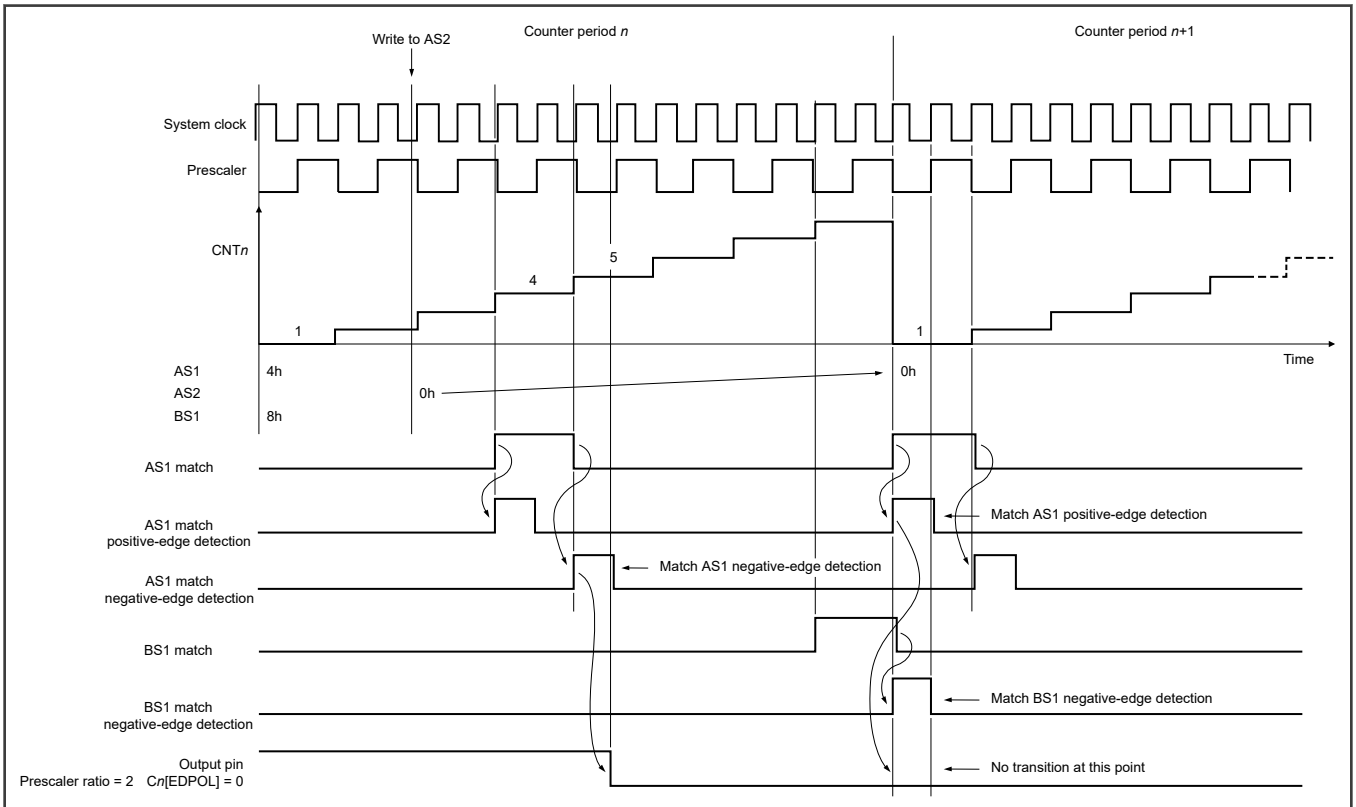


Figure 323. Timing example: AS1 = 0

60.6.3.16.8 Loading AS1 and BS1

As illustrated in Timing example: AS1 and BS1 update and FLAGs, AS1 and BS1 both use the same signal, generated at the last system clock cycle of a counter period. Therefore, eMIOS updates AS1 and BS1 with AS2 and BS2 values, respectively, at the same time that it changes the internal counter (CNT*n*[C]) to 1h. This event is the counter period boundary. The load signal pulse lasts for one system clock cycle. If you write AS2 and BS2 within counter period *n*, their values are available in AS1 and BS1, respectively, at the first clock cycle of counter period *n*+1. The UC then uses these new values for matches at counter period *n*+1. You can use Output Update Disable (OUDIS) to delay updating AS1 and BS1 for synchronization purposes.

60.6.3.16.9 Timing example: AS1 and BS1 update and FLAGs

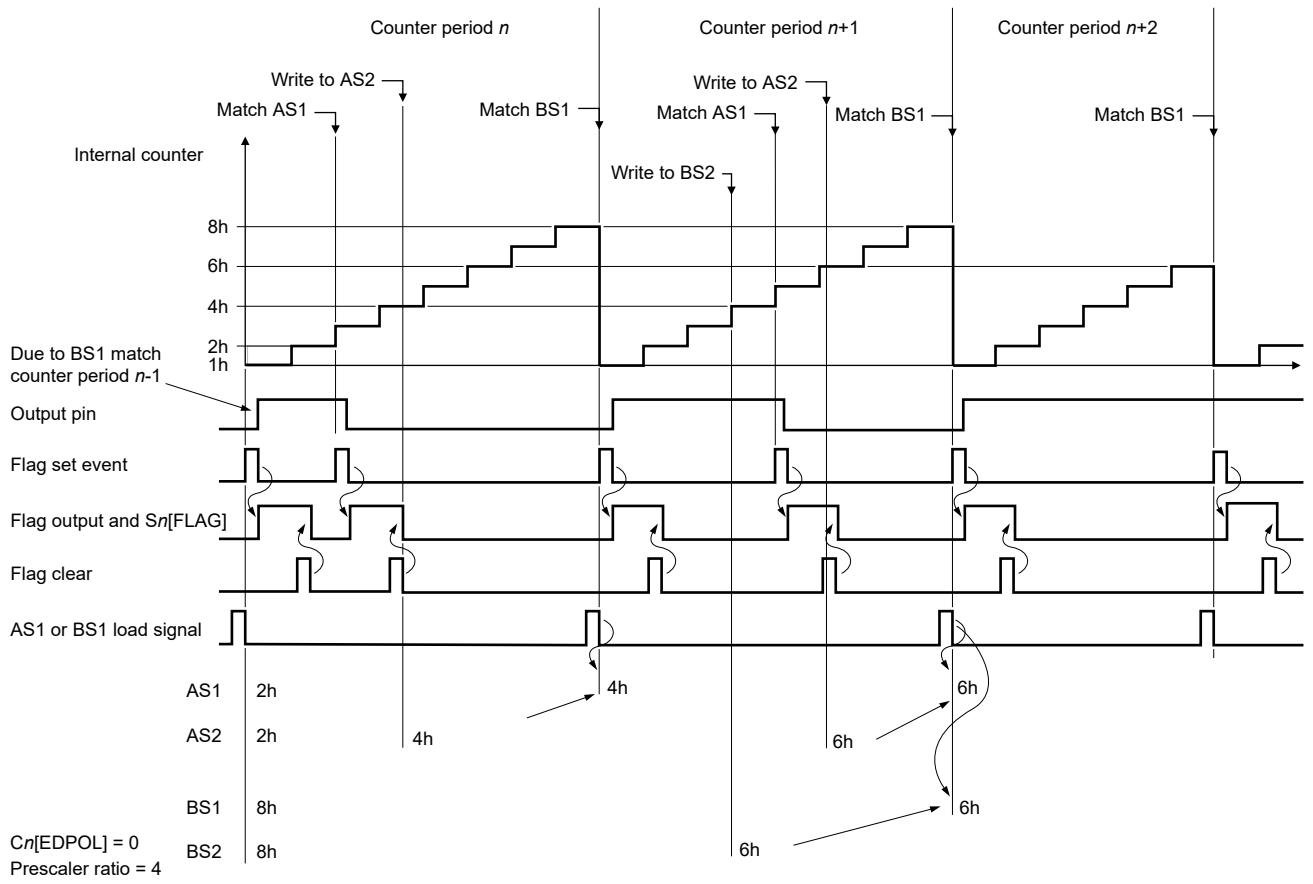


Figure 324. Timing example: AS1 and BS1 update and FLAGs

60.6.3.16.10 Flag generation

Timing example: AS1 and BS1 update and FLAGs assumes that both the channel and global prescalers are 1h (divide ratio of 2), causing the internal counter to transition every four system clock cycles. You can configure flags to be generated only on BS1 matches when bit 1 of C*n*[MODE] is 0, or on both AS1 and BS1 matches when that bit is 1. Because the BS1 flag occurs at the period boundary, you can use this flag to indicate that the AS2 or BS2 data written in period *n* has been loaded to AS1 or BS1, respectively, thereby generating matches in period *n*+1. Flag operation is synchronous, which means the flag sets one system clock cycle after the set-flag event.

60.6.3.16.11 Output disable

As illustrated in Timing example: active output disabled, the output-disable feature in OPWFMB mode forces the channel output flip-flop to the value of C*n*[EDPOL]. This functionality supports applications that use active-high signals and a high-to-low transition at an AS1 match. In this case, you must write 0 to C*n*[EDPOL]. You also must configure the internal counter to transition on every system clock cycle (write 0 to both GCP and CP).

The output-disable feature operates synchronously, meaning that the assertion of the Output Disable input signal causes the channel output flip-flop to transition to C*n*[EDPOL] at the next system clock cycle. If the Output Disable input pin deasserts, the output-signal transition occurs at the next AS1 or BS1 match.

Timing example: active output disabled assumes that the Output Disable input is enabled (using C*n*[ODIS]) and selected (using C*n*[ODISSL]). See UC Control n (C0 - C23) for a detailed description of the ODIS and ODISSL fields that enable and select the Output Disable inputs, respectively.

60.6.3.16.12 Timing example: active output disabled

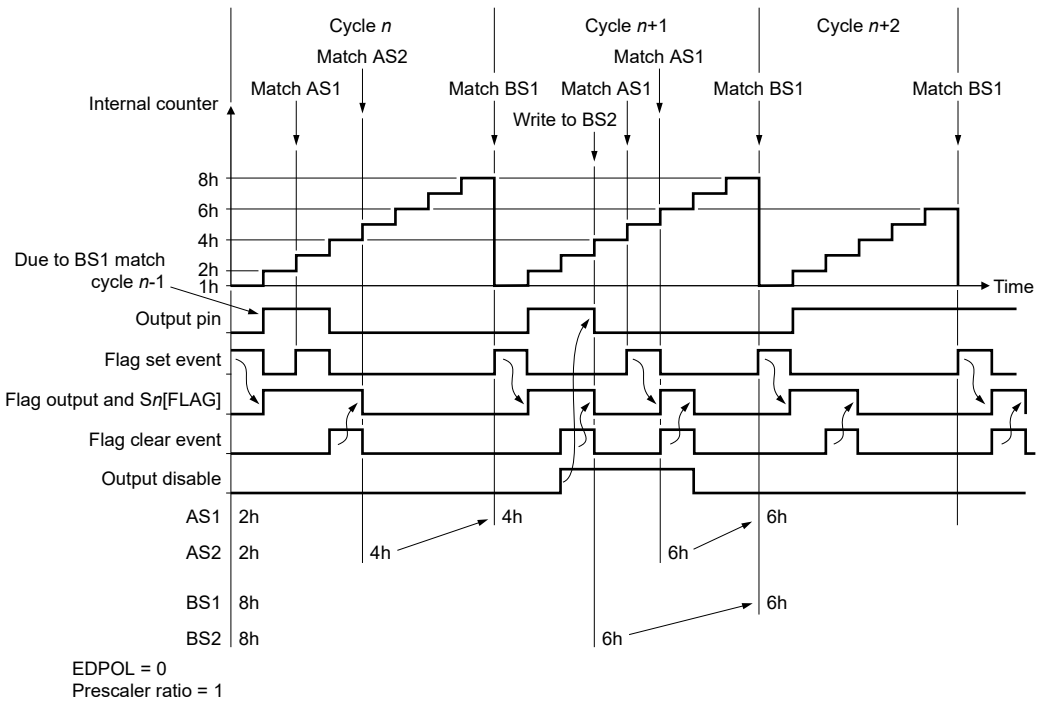


Figure 325. Timing example: active output disabled

60.6.3.16.13 Force match

Force Match A (C*n*[FORCMA]) and Force Match B (C*n*[FORCMB]) allow you to force the output flip-flop to the level corresponding to a match on comparators A or B, respectively. Similarly to a BS1 match, Force Match B changes the internal counter to 1h. The force-match operations do not set the input-capture flag (S*n*[FLAG]).

60.6.3.16.14 100% and 0% duty cycles

Timing example: from 100% to 0% duty cycle illustrates the generation of 100% and 0% duty-cycle signals. Initially, AS1 = 8h and BS1 = 8h. In this case, a BS1 match has precedence over an AS1 match—therefore, the output flip-flop acquires the complement of C*n*[EDPOL]. This configuration corresponds to a 100% duty-cycle signal. The same output signal can be generated for any AS1 value greater than or equal to BS1.

If AS1 is 0, the UC generates a 0% duty-cycle signal. In this case, the BS1 = 8h match from period 8 occurs at the same time as the AS1 = 0h match from period 9. See Timing example: AS1 = 0 for AS1 and BS1 match generation. In this case, an AS1 match has precedence over a BS1 match, and the output signal transitions to C*n*[EDPOL].

60.6.3.16.15 Timing example: from 100% to 0% duty cycle

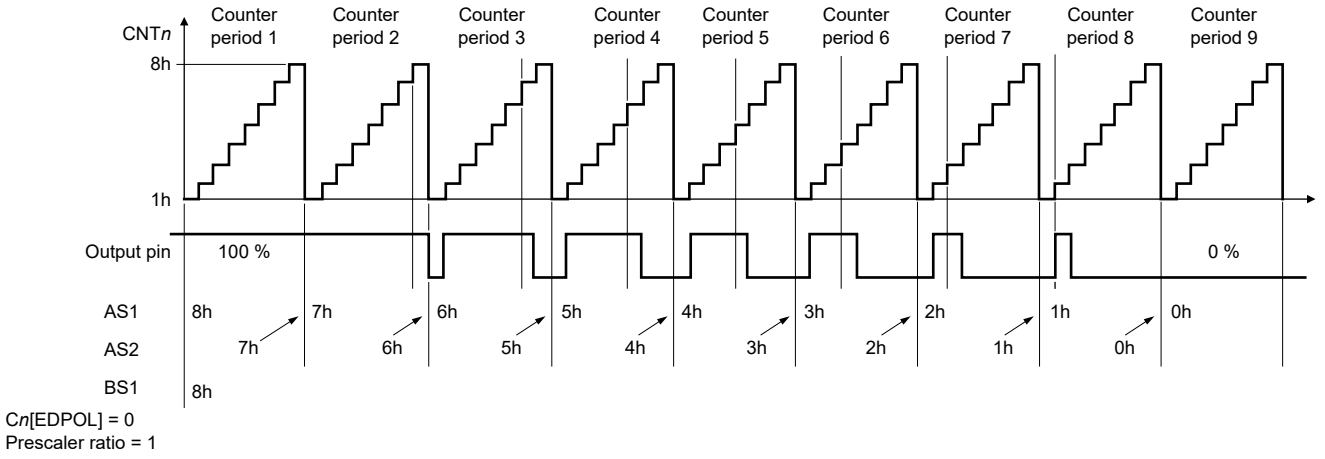


Figure 326. Timing example: from 100% to 0% duty cycle

60.6.3.17 Center Aligned Output PWM with Dead Time Insertion Buffered (OPWMCB) mode

60.6.3.17.1 Overview

In this mode, the UC generates a center-aligned PWM pulse with dead time insertion in the leading or trailing edge. See OPWMCB functions and MODE field values. AS1 and BS1 are double-buffered to allow smooth output signal generation when you change the AS2 or BS2 values.

When the UC enters OPWMCB mode from GPIO mode:

- The UC drives the complement of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.
- Select the timebase using Bus Select (C*n*[BSL]). The timebase you select must be running in Up Count-Down Count mode. See Timing example: Up Count submode flag assertion. You must start the MCB timebase after the UC enters OPWMCB mode to avoid missing a match on the first duty cycle.
- Store the ideal duty cycle for the PWM signal in AS1 (write to A*n*[A]), which the UC compares with the selected timebase.
- Store the dead time value in BS1 (write to B*n*[B]), which the UC compares with the internal counter (CNT*n*[C]).

The internal counter may use the internal prescaler ratio, while the selected timebase may use a different prescaler ratio. The UC always resets the internal counter to 1h when an AS1 match occurs.

As in OPWMB and OPWFMB modes, the UC generates the match signal which compares the selected timebase with AS1 or BS1. When the UC deasserts the channel's combinational comparator output signal, it uses the match signal to assert or negate the channel's output flip-flop. See Timing example: AS1 and BS1 match for an illustration of the delay from match to output flip-flop transition in OPWFMB mode. The operation of OPWMCB mode is similar to OPWFMB mode with respect to match behavior and output pin transition.

See Entering OPWMCB mode.

60.6.3.17.2 OPWMCB functions and MODE field values

Table 341. OPWMCB functions and mode field values

Function	C*n*[MODE] (binary)
Insert trailing edge dead time with duty cycle determined by BS1 + AS1	101_11p0[1]

Table continues on the next page...

Table 341. OPWMCB functions and mode field values (continued)

Function	C*n*[MODE] (binary)
Insert leading edge dead time with duty cycle determined by BS1 – AS1	101_11p1[1]
Assert the input capture flag (Sn[FLAG]) on the trailing edge of the output PWM signal	101_110p[1]
Assert the input capture flag (Sn[FLAG]) on both edges of the output PWM signal	101_111p[1]

1. *p* = Adjust parameters for submodes and options. See Unified channels (UC).

60.6.3.17.3 Entering OPWMCB mode

To enter OPWMCB mode (this procedure assumes the channel is initially in GPIO mode):

1. Disable the GCP (write 0 to MCR[GPREN]).
2. For the timebase (MCB) channel:
   a. Disable the CP (write 0 to C*n*[UCPREN]).
   b. Initialize the internal counter (write 1h to CNT*n*[C]).
   c. Initialize the A data (write appropriate value to A*n*[A]).
   d. Configure the UC for MCB Up Count-Down Count mode (write 101_01*xy*b to C*n*[MODE], where *x* = 0 for flag set on match start or *x* = 1 for flag set at period boundary, and *y* = 0 for internal clock source or *y* = 1 for external clock source).
   e. Select the CP ratio (C*n*[UCPRE]).
   f. Enable the CP (write 1 to C*n*[UCPREN]).
3. For the OPWMCB channel:
   a. Disable the CP (write 0 to C*n*[UCPREN]).
   b. Initialize the A data (write appropriate value to A*n*[A]).
   c. Initialize the B data (write appropriate value to B*n*[B]).
   d. Select the timebase input (C*n*[BSL]).
   e. Configure the UC for MCB Up Count-Down Count mode (write 101_01*xy*b to C*n*[MODE], where *x* = 0 for flag set on match start or *x* = 1 for flag set at period boundary, and *y* = 0 for internal clock source or *y* = 1 for external clock source).
   f. Select the CP ratio (C*n*[UCPRE]).
   g. Enable the CP (write 1 to C*n*[UCPREN]).
4. Enable the GPC (write 1 to MCR[GPREN]).

60.6.3.17.4 Timing example: AS1 and BS1 load

When the selected counter bus transitions from 2h to 1h:

- The UC loads AS1 and BS1 as shown in the following figure.
- This event defines the counter period boundary.

The UC loads the values you write to AS2 or BS2 within period *n* into AS1 or BS1, respectively, and uses them to generate matches in period *n*+1.

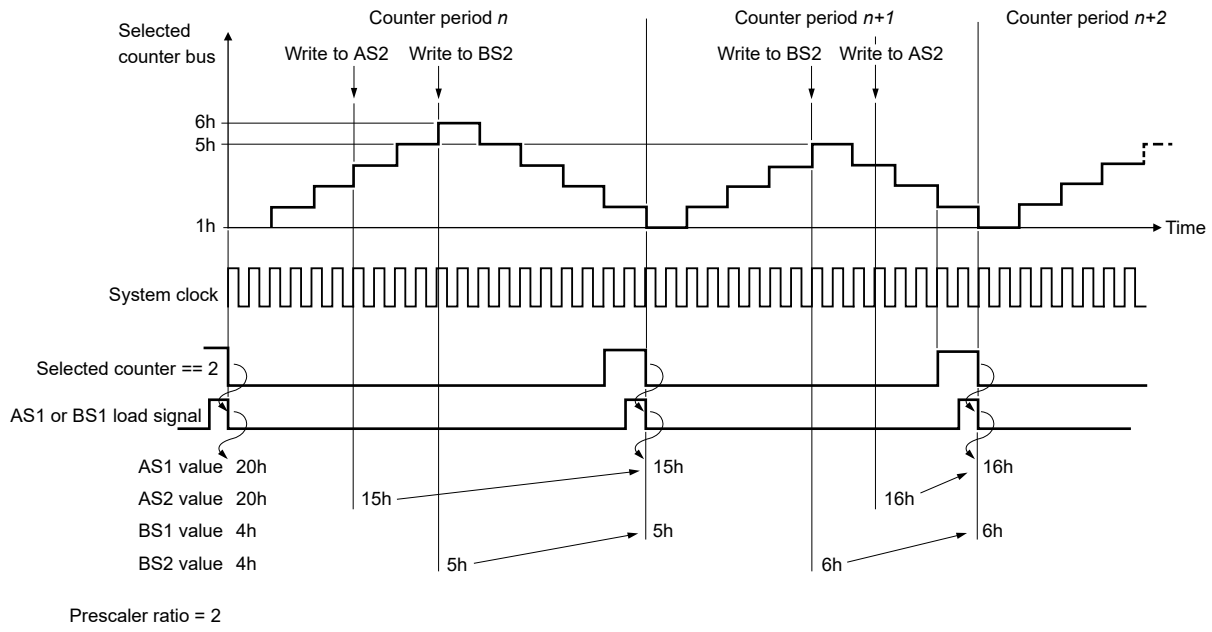


Figure 327. Timing example: AS1 and BS1 load

60.6.3.17.5 Operation sequence with leading edge dead time insertion

When operating with leading edge dead time insertion:

1. When the first AS1 match occurs, the UC writes 1h to the internal counter.
2. When a match between BS1 and the internal timebase occurs, the UC drives the value of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.
3. When the next match between AS1 and the selected timebase occurs, the UC drives the complement of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.

The UC repeats the above sequence continuously.

You must not allow the internal counter to reach 0h as the result of a rollover. To avoid this, you must not write a value greater than

2 x ((external Count Up limit) – A*n*[A])

into B*n*[B].

See Timing example: leading dead time insertion.

60.6.3.17.6 Timing example: leading dead time insertion

This example illustrates two cycles of a center-aligned PWM signal. AS1 and BS1 values change within the same period, which allows you to change the duty cycle and dead time values at the same time.

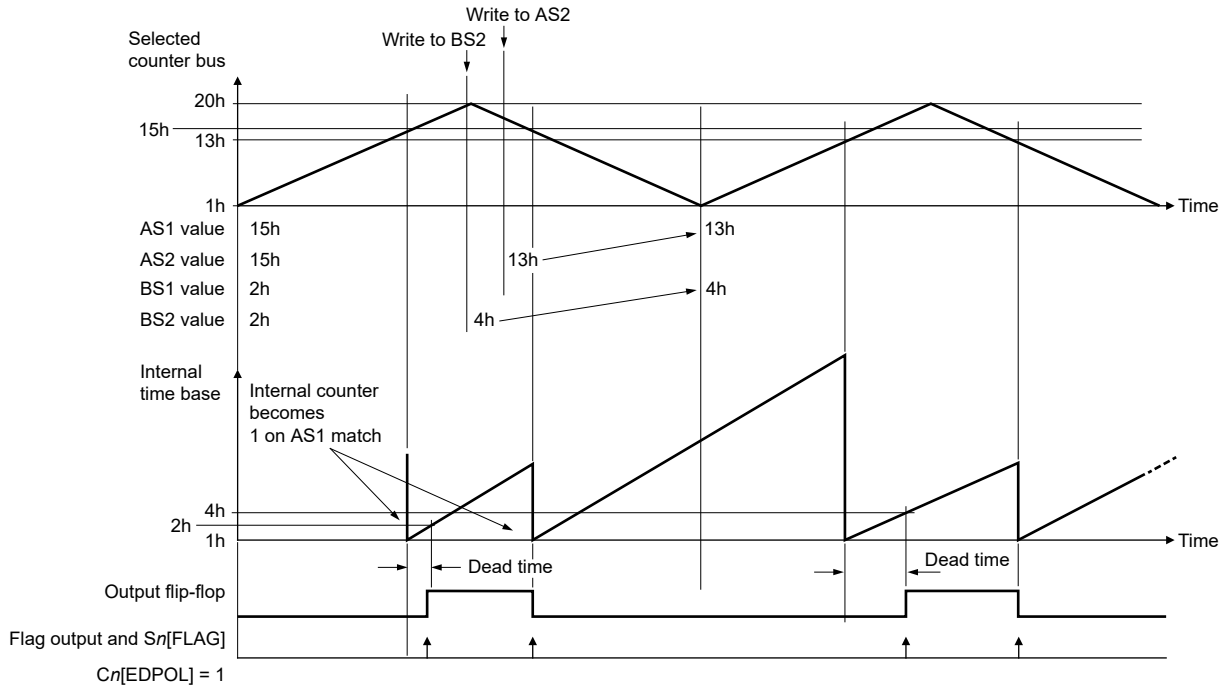


Figure 328. Timing example: leading dead time insertion

60.6.3.17.7 Operation sequence with trailing edge dead time insertion

When operating with trailing edge dead time insertion:

1. When the first match between AS1 and the selected timebase occurs, the UC:
   a. Drives the value of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.
   b. Writes the internal counter (CNT*n*[C]) to 1h.
2. When the second match between AS1 and the selected timebase occurs, the UC:
   a. Writes the internal counter to 1h.
   b. Enables BS1 matches.
3. When a match between BS1 and the selected timebase occurs, the UC drives the the complement of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.

The UC repeats this sequence continuously.

60.6.3.17.8 Timing example: trailing dead time insertion

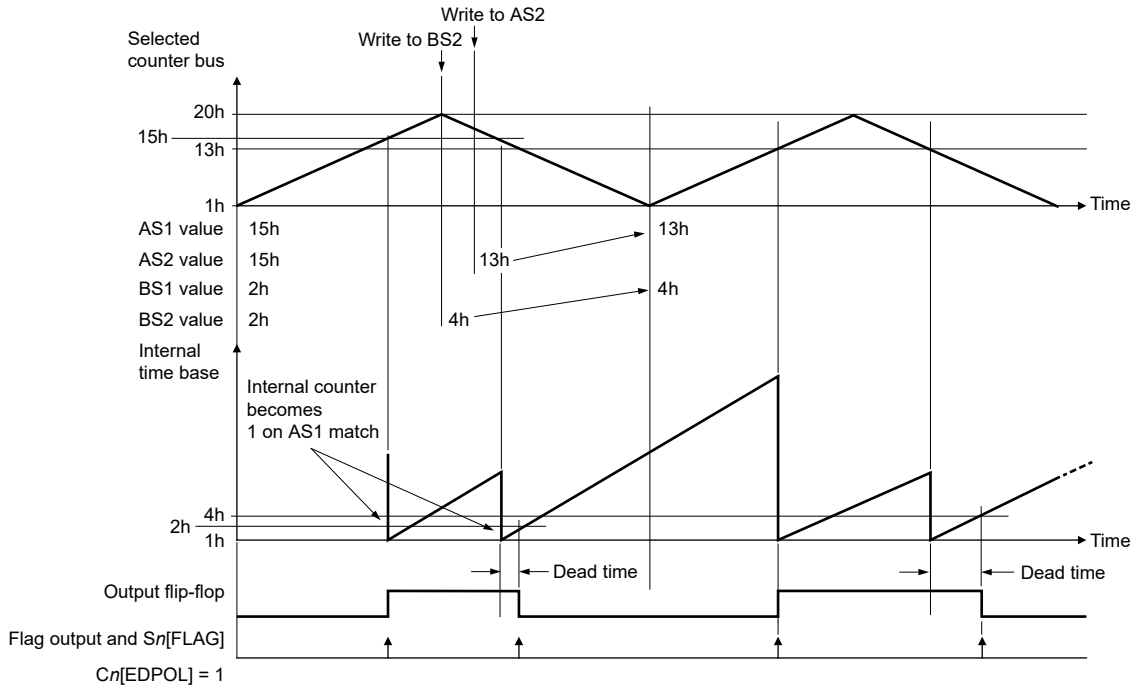


Figure 329. Timing example: trailing dead time insertion

60.6.3.17.9 Force match

Force Match A (C*n*[FORCMA]) and Force Match B (C*n*[FORCMB]) allow you to force the output flip-flop to the level corresponding to a match on comparators A or B, respectively. If subsequent matches occur on comparators A and B, the UC continues to generate PWM pulses regardless of the input capture flag (S*n*[FLAG]). The force-match operations do not set the input-capture flag (S*n*[FLAG]).

When you select leading edge dead time insertion:

- Force Match A drives the complement of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.
- Force Match B drives the value of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.

When you select trailing edge dead time insertion:

- Force Match A drives the value of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.
- Force Match B drives the complement of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.

Force Match operation does not write 1h to the internal timebase.

Force Match operation does not set the input-capture flag.

When you assert Force Match A and Force Match B at the same time, the UC drives the complement of Edge Polarity (C*n*[EDPOL]) on the output flip-flop. This is the equivalent of saying that Force Match A has precedence over Force Match B when you select leading edge dead time insertion, and Force Match B has precedence over Force Match A when you select trailing edge dead time insertion.

Force Match A and Force Match B have the same output transition behavior in both Freeze state and unfrozen state.

60.6.3.17.10 Duty cycle

You generate a duty cycle from 0% to 100% by writing appropriate values to AS1 and BS1 relative to the period of the external timebase.

To generate a 100% duty cycle waveform:

- Select trailing edge dead time insertion (see OPWMCB functions and MODE field values).
- Write 1h to Edge Polarity (C*n*[EDPOL]).
- The BS1 match must occur on or after the counter period boundary (external counter = 1).

You can also generate a 100% duty cycle waveform by writing 1h to AS1. However, if you do this before the UC enters OPWMCB mode, a 100% duty cycle may not occur in the first PWM period depending upon the pin state at mode entry.

To generate a 0% duty cycle waveform:

- AS1 must be greater than the maximum value of the selected counter period.
- The pin must start the current period with the complement value of Edge Polarity (C*n*[EDPOL]). If starting at 100% duty cycle, you can change Edge Polarity (C*n*[EDPOL]) to its complement by forcing the pin (see Force match).

You must write only non-zero values to AS1. If you write the maximum value of the external counter given by

(external counter period)/2

to AS1, the UC constantly drives the value of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.

UC logic prevents matches from one period propagating to the next period. In trailing edge dead time insertion, the UC masks out BS1 matches that occur late in period *n*, so matches in period *n*+1 are unaffected and the output flip-flop does not transition.

60.6.3.17.11 Timing example: duty cycle

To generate a 100% duty cycle output waveform, write 4h to AS1 and 3h to BS1, as illustrated in this diagram. In this case, the trailing edge is at the boundary of counter period *n*+1, which actually belongs to period *n*+2 and therefore does not cause the output flip-flip to transition.

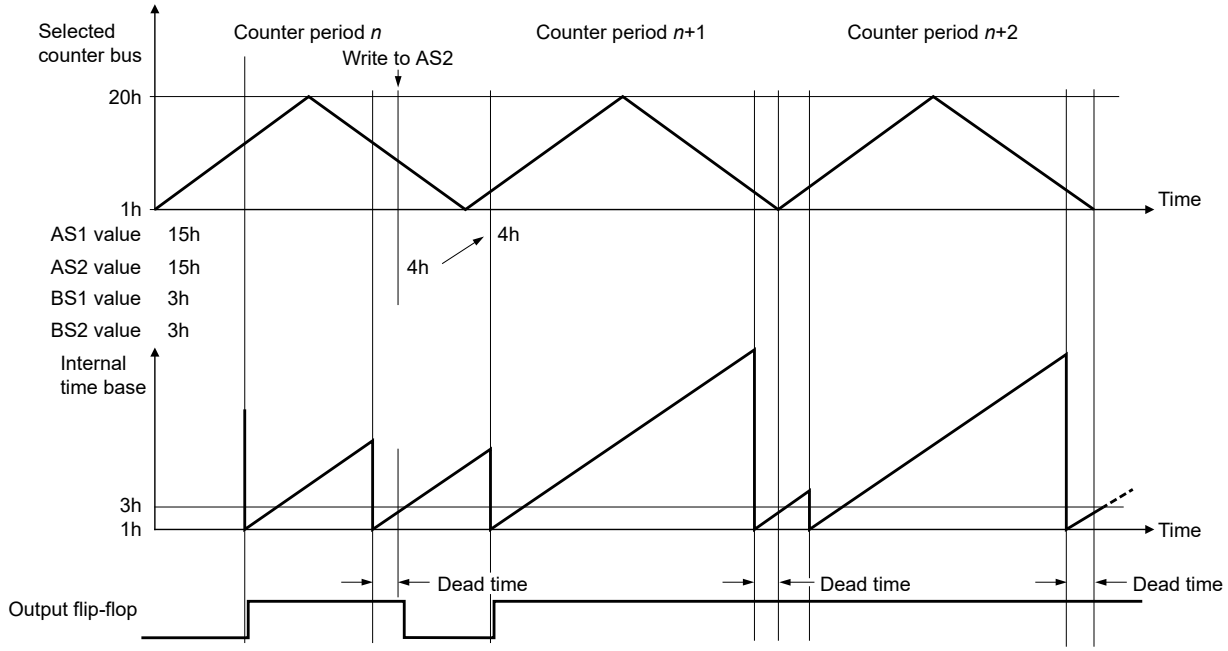


Figure 330. Timing example: duty cycle

60.6.3.17.12 Output disable

You can use Output Update Disable (write 0 to OUDIS[OU*n*]) to disable AS1 and BS1 updates. This allows you to update AS1 and BS1 in the same counter period and synchronize the loading of these registers with the loading of AS1 or BS1 in other channels.

Asserting Output Update Disable drives the complement of Edge Polarity (C*n*[EDPOL]) on output flip-flop. The internal channel matches continue to occur, and therefore the UC continues to set flags.

When you deassert Output Update Disable, the output flip-flop is again controlled by AS1 and BS1 matches. The UC transitions the output flip-flop only on system clock edges.

60.6.3.18 Output PWM Buffered (OPWMB) mode

60.6.3.18.1 Overview

In this mode, the UC produces PWM pulses with programmable leading and trailing edge placement. You must select an external counter driven in MCB Up mode from one of the counter buses. AS1 and BS1 define the first and second edges, respectively. C*n*[EDPOL] defines the output signal polarity. If C*n*[EDPOL] = 0, a negative edge occurs when AS1 matches the selected counter bus and a positive edge occurs when BS1 matches the selected counter bus.

See OPWMB submodes and MODE field values for the values that you must write to C*n*[MODE] to enter this mode.

AS1 and BS1 are double-buffered and updated from AS2 and BS2, respectively, at the cycle boundary. The load operation is similar to the one that occurs in OPWFMB mode. See Timing example: AS1 and BS1 update and FLAGs for more information about AS1 and BS1 updates.

When the UC enters OPWMB mode, the UC drives C*n*[EDPOL] on the output flip-flop.

These rules apply to OPWMB mode:

- If the AS1 and BS1 matches occur at the same time within the same counter period, the BS1 match has precedence over the AS1 match.
- An AS1 = 0 match in period *n* has precedence over a BS1 match in period *n*-1.
- The UC masks out AS1 matches if they occur after a BS1 match within the same period.
- The UC loads any value that you write to AS2 or BS2 in period *n* into AS1 and BS1 at the following period boundary (assuming that the corresponding OU*n* field of Output Update Disable (OUDIS) is 0). Therefore, the UC uses the new values for AS1 and BS1 matches in period *n*+1.

Timing example: 100% to 0% duty cycle illustrates a waveform changing from 100% to 0% duty cycle. In this example, BS1 is programmed to the same value as the period of the external selected timebase. If you write a value smaller than 8h to BS1, you cannot achieve a 0% duty cycle by only changing AS1. Because BS1 matches have precedence over AS1 matches, the flag output transitions to the complement of C*n*[EDPOL] at a BS1 match. For another example, if you write 9h to BS1, a BS1 match does not occur, and the UC generates a 0% duty-cycle signal.

60.6.3.18.2 OPWMB submodes and MODE field values

Table 342. OPWMB submodes and MODE field values

When FLAG generation occurs	C*n*[MODE] (binary)
At a BS1 match	110_0000
At an AS1 or BS1 match	110_0010

60.6.3.18.3 Force match

Force Match A (C*n*[FORCMA]) and Force Match B (C*n*[FORCMB]) allow you to force the output flip-flop to the level corresponding to a match on comparators A or B, respectively. If subsequent matches occur on comparators A and B, the UC continues to generate PWM pulses regardless of the input capture flag (S*n*[FLAG]). The force-match operations do not set the input-capture flag (S*n*[FLAG]).

60.6.3.18.4 Matches and flag operation

Timing example: matches and flags illustrates the operation of OPWMB mode with an emphasis on AS1 and BS1 matches and the transition of the channel output pin. The output pin transitions are based on the negative edges of the AS1 and BS1 match signals. AS1 becomes zero in period *n*+1. In this case, the UC uses the match positive edge instead of the negative edge to transition the output flip-flop.

60.6.3.18.5 Timing example: matches and flags

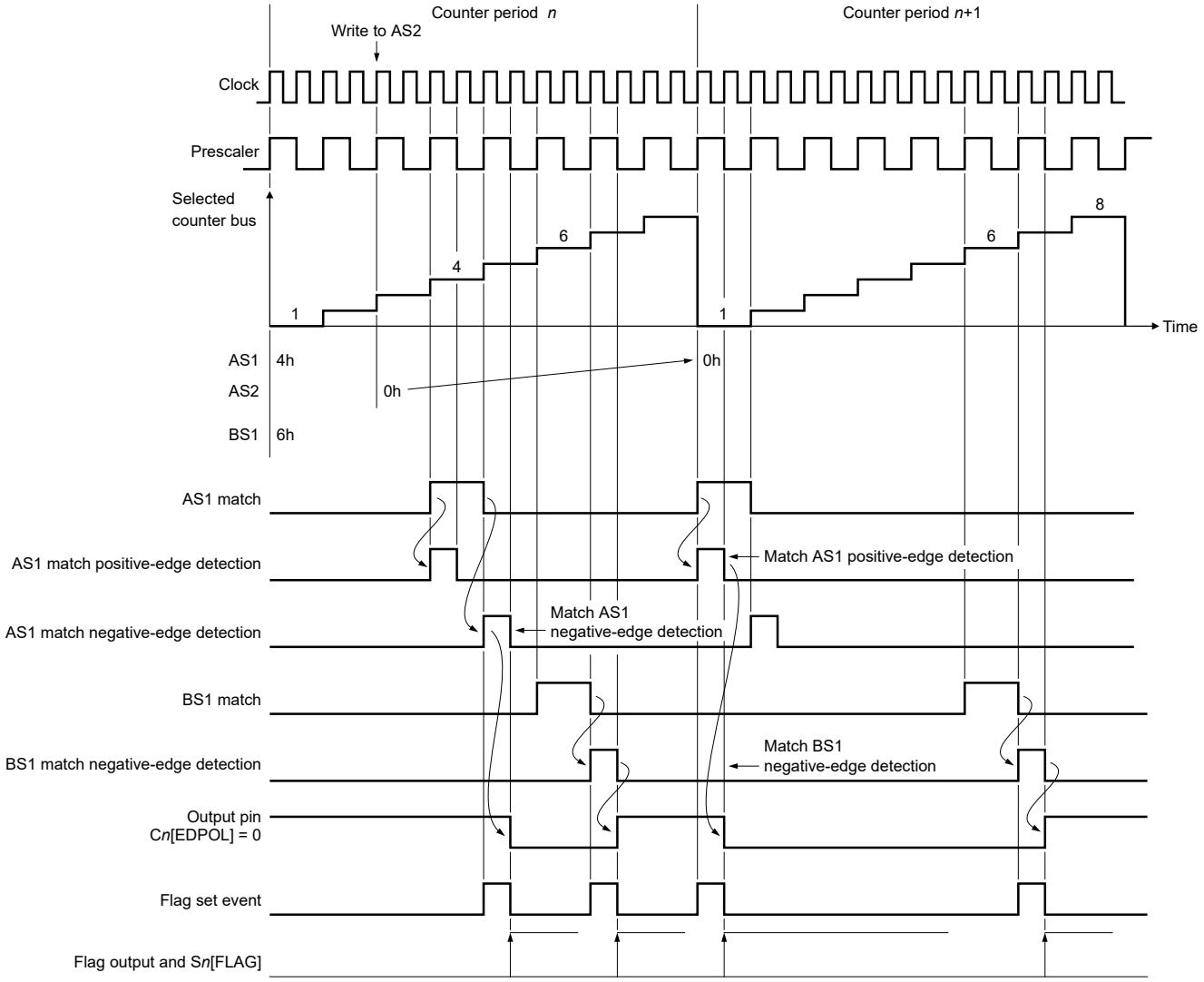


Figure 331. Timing example: matches and flags

60.6.3.18.6 0% duty cycle

Timing example: 0% duty cycle illustrates the channel operation for a 0% duty cycle. The AS1 match positive-edge signal occurs at the same time as the BS1 = 8h negative-edge signal. In this case, the AS1 match has precedence over the BS1 match, causing the output pin to remain at the value of C*n*[EDPOL] and generating a 0% duty-cycle signal.

60.6.3.18.7 Timing example: 0% duty cycle

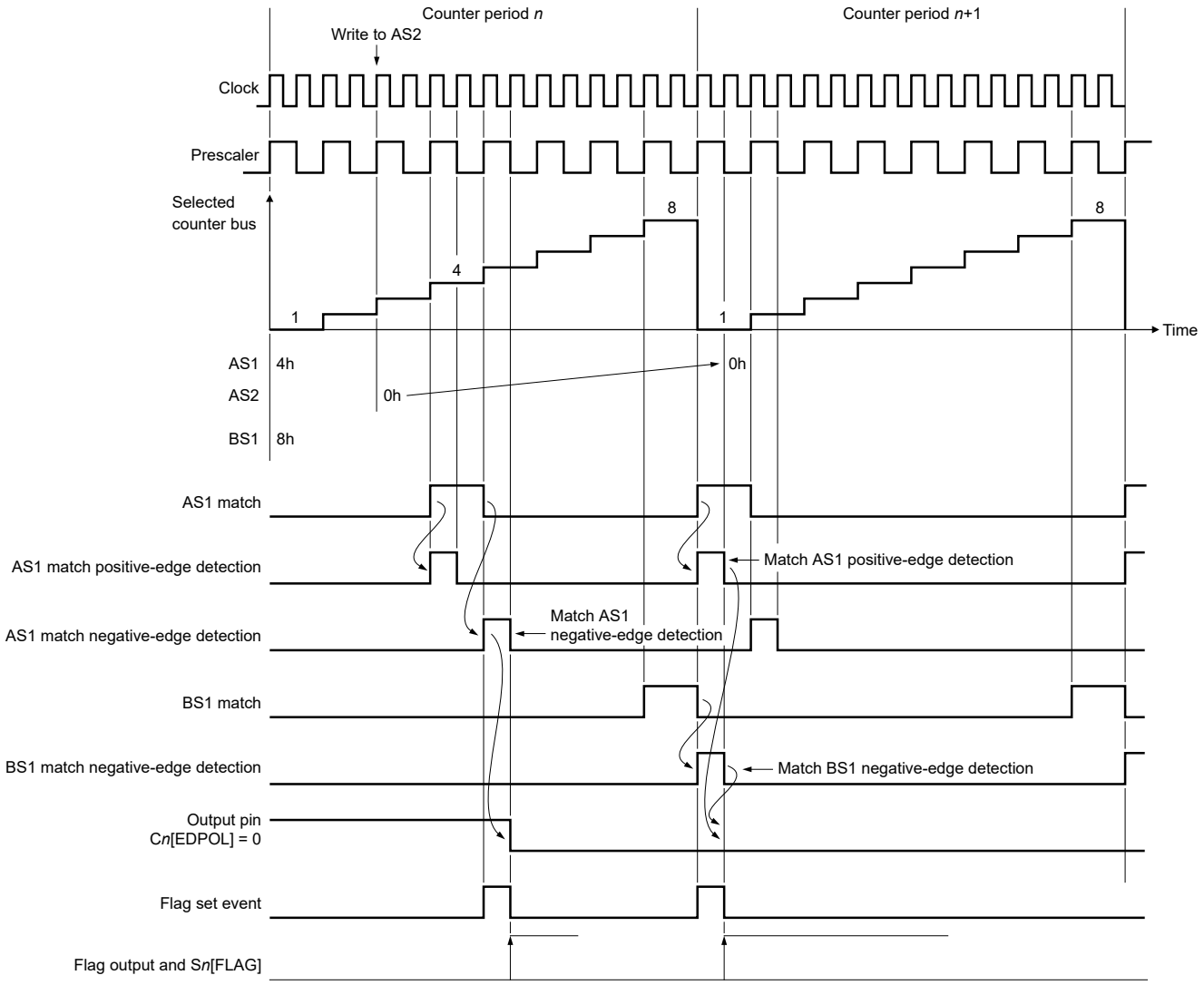


Figure 332. Timing example: 0% duty cycle

60.6.3.18.8 Active output disabled

Timing example: active output disabled illustrates the operation of the OPWMB mode with an emphasis on the assertion of the output-disable signal. The output disable forces a transition in the output pin to the value of C*n*[EDPOL]. The deassertion of the output-disable signal allows the output pin to transition at the following AS1 or BS1 match. The output disable does not modify the behavior of S*n*[FLAG]. A delay of one system clock cycle exists between the assertion of the output-disable signal and the transition of the output pin to C*n*[EDPOL].

60.6.3.18.9 Timing example: active output disabled

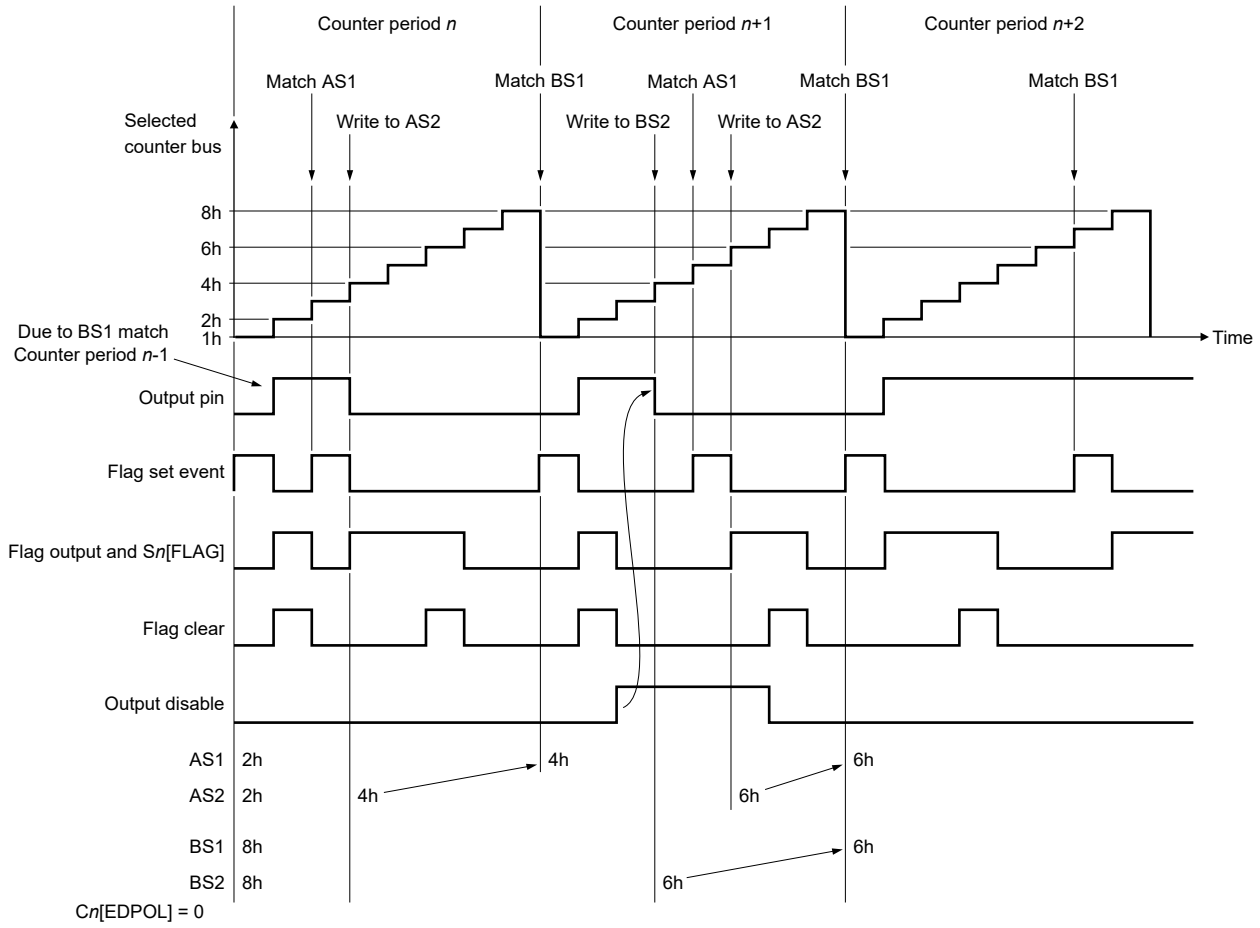


Figure 333. Timing example: active output disabled

60.6.3.18.10 Changing from 100% to 0% duty cycle

Timing example: 100% to 0% duty cycle illustrates a waveform changing from 100% to 0% duty cycle. In this example, BS1 has the same value as the period of the external selected timebase. If you write a value smaller than 8h to BS1, you cannot achieve a 0% duty cycle by only changing AS1. Because BS1 matches have precedence over AS1 matches, the output pin transitions to the complement of C*n*[EDPOL] at a BS1 match. For another example, if you write 9h to BS1, a BS1 match does not occur, and the UC generates a 0% duty-cycle signal.

60.6.3.18.11 Timing example: 100% to 0% duty cycle

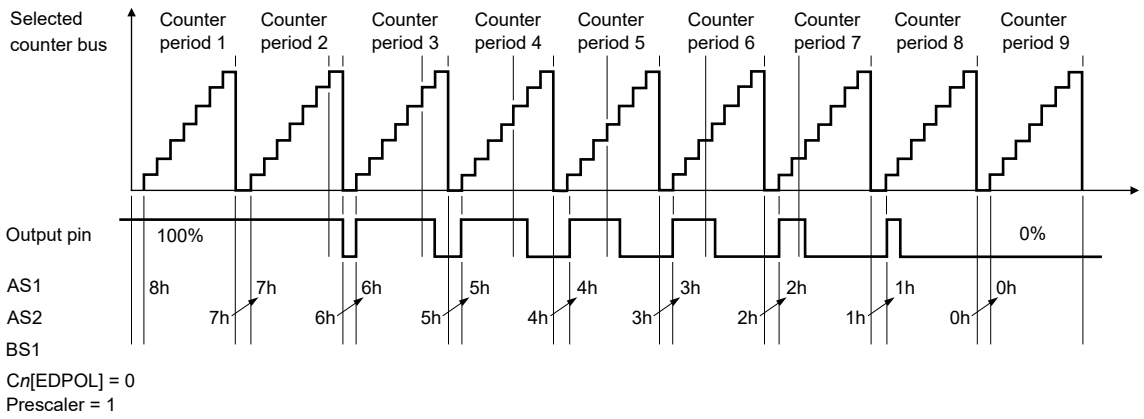


Figure 334. Timing example: 100% to 0% duty cycle

60.6.3.19 Output PWM with Trigger (OPWMT) mode

60.6.3.19.1 Overview

In this mode, the UC generates PWM pulses where the period does not change while the UC outputs the signal, but the duty cycle changes. When the duty cycle changes, it must not create glitches. The mode supports multiple UCs executing in the same mode and sharing a common timebase. Each UC can have a fixed PWM leading-edge position with respect to the other UCs. It can also generate a trigger signal at any point in the period. eMIOS can output this trigger signal to initiate activity in other parts of the chip—for example, to start ADC conversions.

You must select an external counter, driven in either MC Up or MCB Up mode (see Modulus Counter Buffered (MCB) mode), from one of the counter buses.

AS1 in OPWMT mode, AS2 in OPWMT mode, and BS1 and BS2 in OPWMT mode describe the configuration and behavior of the individual shadow registers in this mode.

To account for the shift in the AS1-defined leading edge of the waveform, the BS1-defined trailing edge may roll over into the next period. This means that a match against BS1 does not have to be qualified by a match in AS1. This also means that if you incorrectly write a value less than AS1 to BS1, the output persists over a period boundary until the UC encounters the BS1 value.

This mode provides a buffered update of the trailing edge by updating BS1 with BS2 contents only at a match of AS1.

The positive edges of the AS1, BS1, and AS2 match signals drive the output pin and flag transitions. See Timing example: matches and flags for details on positive-edge matches.

The UC sets the input capture flag (S*n*[FLAG]) only at an AS2 match. An AS1, BS1, or BS2 match has no effect on the flag.

When the UC enters OPWMT mode, it drives the complement of C*n*[EDPOL] on the output flip-flop.

Timing example: OPWMT illustrates the UC running in OPWMT mode with trigger-event generation and duty-cycle update after the next-period update.

60.6.3.19.2 Timing example: OPWMT

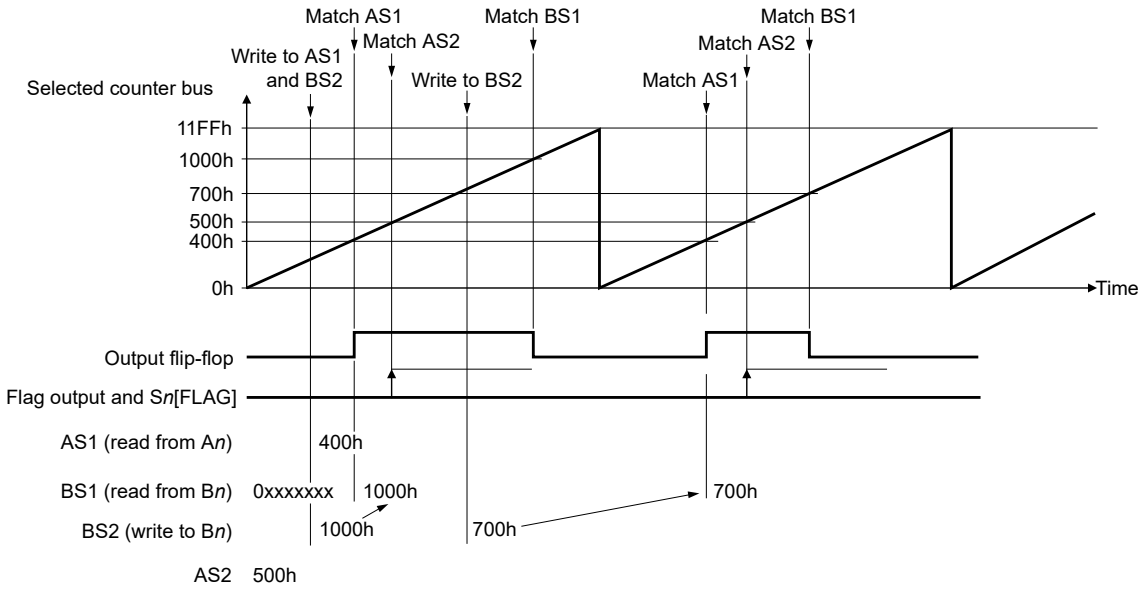


Figure 335. Timing example: OPWMT

60.6.3.19.3 AS1 in OPWMT mode

AS1 defines the leading edge of the PWM pulse output, and therefore the beginning of the PWM period. Using this fact, you can ensure that, when using a shared timebase, the leading edges of multiple UCs in OPWMT mode occur at specific times with respect to the other UCs. You can also introduce a fixed offset for each UC, which is particularly useful in the generation of lighting control signals. For this application, to reduce or eliminate noise generation, you can shift the PWM pulse of each UC with respect to the selected timebase by adjusting the AS1 value for each channel. The value you write to AS1 must be within the range of the selected timebase. Shadow registers (AS1, AS2, BS1, and BS2) loaded with 0h do not produce matches if the channel driving the timebase is in MCB mode.

AS1 is not buffered, because the shift of a PWM channel must not change while the UC generates the PWM signal. If you modify AS1, it updates immediately and one PWM pulse could be lost.

The UC compares AS1 to the selected timebase. When a match on AS1 occurs, the UC drives C*n*[EDPOL] on the output flip-flop. When a match occurs on comparator B, the UC drives the complement of C*n*[EDPOL] on the output flip-flop.

60.6.3.19.4 AS2 in OPWMT mode

AS2 defines the generation of a trigger event within the PWM period. The value you write to AS2 must be within the range of the selected timebase—otherwise the UC does not generate a trigger. A match on the comparator asserts the input-capture flag, but the match has no effect on the PWM output signal generation. The typical setup to obtain a trigger with a flag is to enable DMA and drive the DMA acknowledgement or DMA completion input high.

AS2 is not buffered, and therefore the UC updates it immediately. If the UC is running when you make a change, this could cause either of the following:

- The loss of one trigger event
- The generation of two trigger events within the same period

You access AS2 by reading or writing ALTA*n*[ALTA].

60.6.3.19.5 BS1 and BS2 in OPWMT mode

You access BS1 and BS2 using B*n*[B]:

- When you read B*n*[B], you read BS1.
- When you write to B*n*[B], you write to BS2.

BS1 defines the trailing edge of the PWM pulse output, and therefore the duty cycle of the PWM signal. To synchronize the BS1 update with the PWM signal and to ensure a correct output pulse generation, the BS2→BS1 transfer occurs at every AS1 match. This behavior is the same as in OPWM mode with next-period update.

Output Update Disable (OUDIS) affects transfers between BS2 and BS1 only.

60.6.3.19.6 Force match

At any time, Force Match A (C*n*[FORCMA]) and Force Match B (C*n*[FORCMB]) allow you to force the output flip-flop to the level corresponding to a match on AS1 or BS1, respectively. Similarly to a BS1 match, Force Match B changes the internal counter to 1h. The force-match operations do not set the input-capture flag (S*n*[FLAG]). Any match resulting from writing 1 to FORCMA or FORCMB has priority over any simultaneous match regarding output pin transitions. BS2→BS1 transfer at an A match is not inhibited by writing 1 simultaneously to both FORCMA or FORCMB. If you write 1 to both FORCMA and FORCMB simultaneously, the output pin acquires the complement of C*n*[EDPOL], as if AS1 and BS1 had the same value. FORCMA assertion causes the UC to process the BS2→BS1 transfer as a regular A match comparison and not due to FORCMA assertion, regardless of FORCMB assertion. If subsequent matches occur on comparators AS1 and B, the UC continues to generate PWM pulses, regardless of the state of S*n*[FLAG].

60.6.3.19.7 Duty cycle

To achieve 0% duty cycle, you must write the same value to AS1 and BS2. When a simultaneous match on comparators A and B occurs, the UC drives the complement of C*n*[EDPOL] on the output flip-flop at every period.

To achieve 100% duty cycle, you must write a value to BS1 that is greater than the maximum value of the selected timebase. The maximum counter value for the timebase is FF_FFFEh for a 24-bit counter and FFFEh for a 16-bit counter. When a match on comparator AS1 occurs, the UC drives C*n*[EDPOL] on the output flip-flop at every period. The match at comparator A still triggers the BS2→BS1 transfer.

Timing example: 0% duty cycle illustrates the UC running in OPWMT mode with trigger-event generation and a 0% duty cycle.

Timing example: 100% duty cycle illustrates the UC running in OPWMT mode with trigger-event generation and a 100% duty cycle.

60.6.3.19.8 Timing example: 0% duty cycle

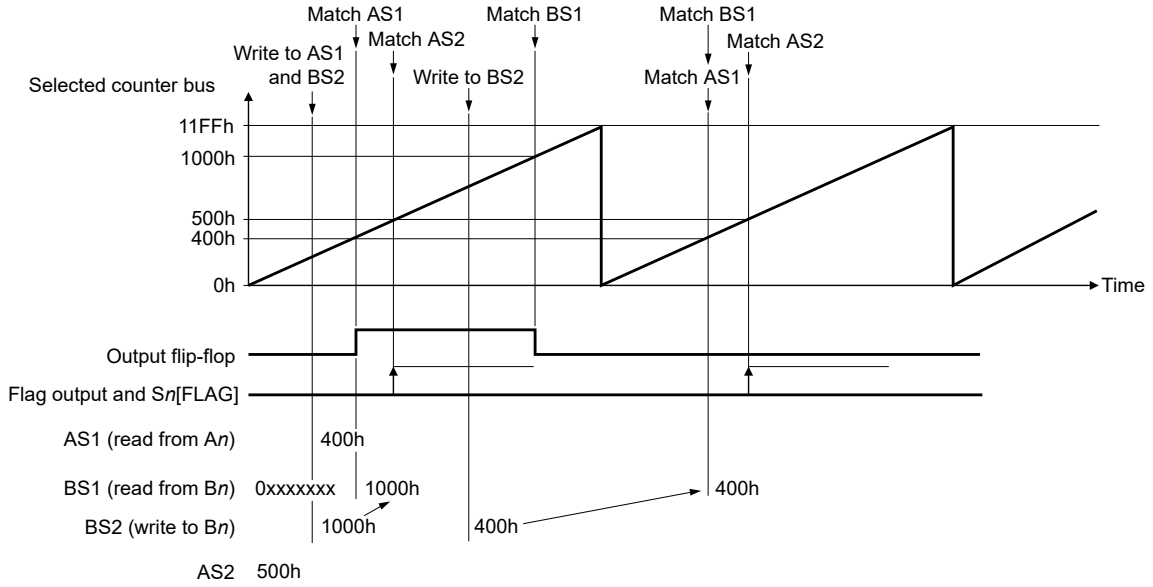


Figure 336. Timing example: 0% duty cycle

60.6.3.19.9 Timing example: 100% duty cycle

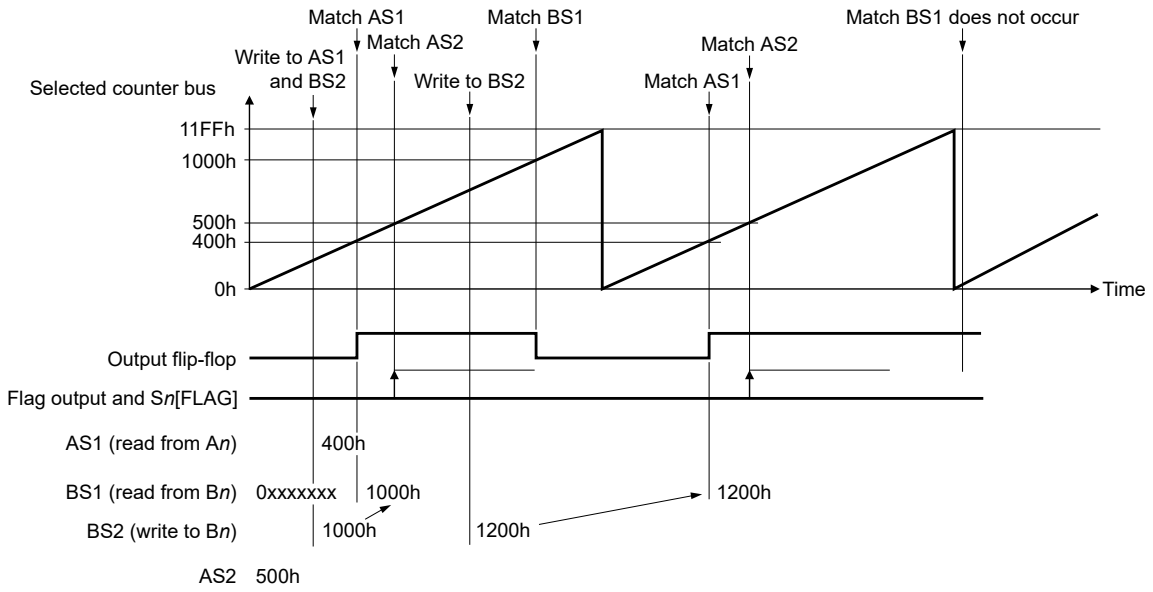


Figure 337. Timing example: 100% duty cycle

60.6.3.19.10 Output disable

As with other UC modes, OPWMT mode implements the output-disable function. When C*n*[ODIS] = 1, assertion of the output-disable input causes the UC to drive C*n*[EDPOL] on the output. Other than the fixed output, the UC continues to operate normally. When the output-disable signal deasserts, the UC returns to full normal operation.

60.6.3.20 Programmable input filter (PIF)

60.6.3.20.1 Overview

The PIF specifies the minimum input pulse width, in clock cycles, that can pass through the filter. It supports bypass operation (C*n*[IF]), with the input signal synchronized before arriving at the digital filter, or from 2 to 16 clock cycles in powers of 2.

60.6.3.20.2 Counter

The PIF is a 5-bit programmable up counter. The selected clock source increments the up counter after the number of clock cycles specified by the input filter (C*n*[IF]0.

The system clock synchronizes the input signal. When this signal changes state, the counter begins incrementing. As long as the new signal state is stable, the counter continues incrementing. If the counter overflows, eMIOS validates the new signal value. In this case, it transmits the value as a pulse edge to the edge detector. If the opposite edge appears on the signal before validation (overflow), eMIOS resets the counter. At the next transition, the counter starts incrementing again. A glitch does not occur on the edge detector for any pulse shorter than the full range of the masked counter.

Neither Freeze state nor deasserted GTBE disables the filter.

60.6.3.20.3 UC PIF logic

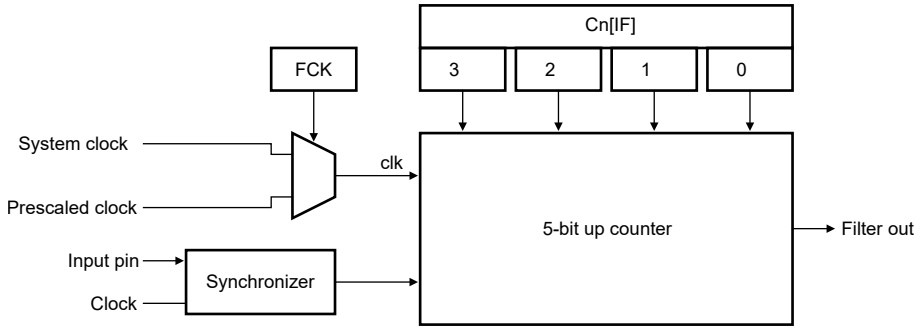


Figure 338. UC PIF logic

60.6.3.20.4 Timing example: UC PIF at 4 cycles

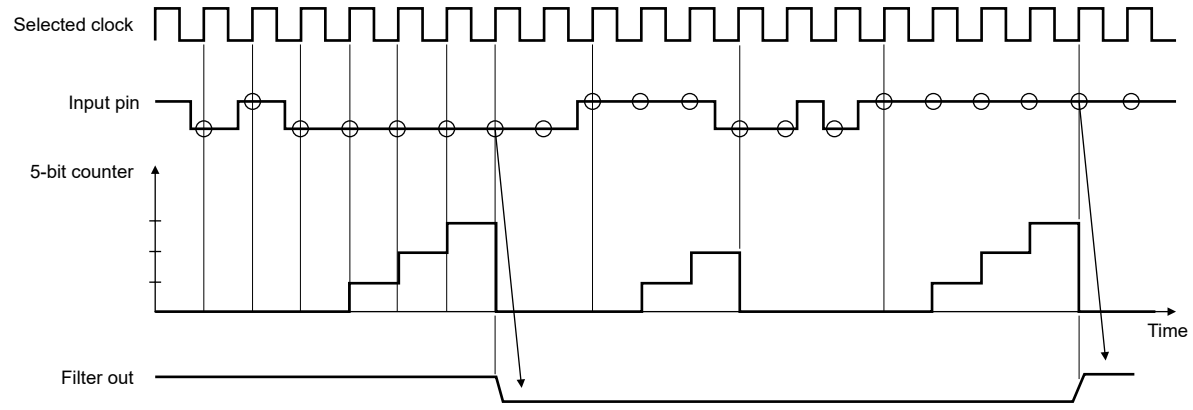


Figure 339. Timing example: UC PIF at 4 cycles

60.6.3.21 UC clock prescaler (CP)

eMIOS divides the GCP output signal by the UC CP (C*n*[UCPRE] + 1) to generate a clock enable for the UC internal counter. See Global clock prescaler (GCP). You enable the UC CP by writing 1 to C*n*[UCPREN]. If you disable the CP (write 0 to the same field), eMIOS stops the UC internal counter.

To ensure safe operation and avoid glitches, you must perform the following steps to change the UC CP:

1. Disable the GCP (write 0 to MCR[GPREN]).
2. Disable the UC CP (write 0 to C*n*[UCPREN]).
3. Specify the new prescaler value (write CP – 1 to C*n*[GPRE].
4. Enable the UC CP (write 1 to C*n*[UCPREN]).
5. Enable the GCP (write 1 to MCR[GPREN]).

60.6.4 Peripheral bus interface unit (BIU)

The BIU provides the interface between the IIB and the peripheral bus, which allows communication among all submodules and the IP interface. The BIU supports 8-, 16-, and 32-bit accesses performed over a 32-bit data bus in a single clock cycle.

60.6.5 Global clock prescaler (GCP)

eMIOS divides the system clock by the GCP (MCR[GPRE] + 1) and routes the resulting prescaled clock output to the channel CPs. You enable the GCP by writing 1 to MCR[GPREN]. If you disable the GCP (write 0 to the same field), eMIOS clears the prescaler counter and stops the prescaler clock.

To ensure safe operation and avoid glitches, you must perform the following steps to change the GCP:

1. Disable the GCP (write 0 to MCR[GPREN]).
2. Specify the new prescaler value (write GCP – 1 to MCR[GPRE].
3. Enable the GCP (write 1 to MCR[GPREN]).

60.6.6 Freeze and chip Debug mode

60.6.6.1 Enter and exit Freeze state

When the chip is in Debug mode (the chip Debug signal is asserted), you can freeze individual UCs. For the effects of Freeze state on eMIOS functions, see Freeze effects.

To put a channel in Freeze state, you must meet all of the following conditions:

- The chip must assert Debug.
- Prepare eMIOS for the Freeze state (write 1 to MCR[FRZ]).
- Enable Freeze for each channel you want to freeze:
  - Write 1 to C*n*[FREN].

A channel exits Freeze state when any of the following conditions occur:

- The chip deasserts Debug.
- You take eMIOS out of the Freeze state (write 0 to MCR[FRZ]).
- You disable Freeze for the channel:
  - write 0 to C*n*[FREN].

60.6.6.2 Freeze effects

Function	Effects of freeze
GCP	No effect
UC	• Halts counter, capture, and compare operation. • Freezes UC in its current state. • Makes all registers accessible. • In output modes, force match remains operational, allowing software to force the output to the desired level. • In input modes, the UC ignores input events. • When eMIOS exits Freeze state then UC operations resume, but they may be inconsistent until the UC reenters GPIO mode.
STAC	No effect
BIU	No effect

60.7 Initialization information

When initializing eMIOS, take the following considerations into account.

For eMIOS behavior on reset, see Reset.

Before changing the UC operating mode, you must:

1. Put the UC in GPIO mode (write 0b or 1b to C*n*[MODE]).
2. Write the correct values for the next operating mode to A*n* and B*n*.
3. Put the UC in the new operating mode (write the appropriate value to C*n*[MODE]).

If you change a UC from one mode to another without performing this procedure, the first operation cycle of the selected timebase can be random; that is, matches can occur in random time because you did not update A*n* and B*n* with correct values before the timebase matches the previous contents of either register.

NOTE

To to avoid causing an unexpected interrupt or DMA request when changing from GPIO to MCB mode, you must use the procedure described in Changing to MCB mode.

When you enable interrupts, any interrupt service routine must clear the flag before exiting.

60.8 Application information

60.8.1 Overview

All output operation modes can generate correlated output signals. You can disable updates of the output signals for each UC (OUDIS).

To guarantee that incrementing of the internal counters of correlated channels occurs in the same clock cycle, you must configure the internal CPs before enabling the GCP. If you configure the internal CPs after enabling the GCP, the internal counters may increment in the same ratio, but at different clock cycles.

You should drive the Output Disable Input signals with the flag signals of some UCs running in SAIC mode. When an output disable condition happens, the interrupt service routine must process the output channels before processing the channels running SAIC. This sequence avoids glitches in the output pins.

60.8.2 Timebase generation

In the MC and OPWFM modes with internal clock sources, you can modify the internal counter rate by configuring the CP ratio. Timebase with the fastest prescaler ratio illustrates a timebase with a prescaler ratio of one.

NOTE

The buffered modes, MCB and OPWFMB, behave differently.

If the prescaler ratio is greater than one or you select an external clock, the internal counter behaves in one of four different ways depending on the channel mode:

- In MC mode with Clear on Match Start and External Clock: Timebase generation with clear on match start and external clock.
- In MC mode with Clear on Match Start and Internal Clock source: Timebase generation with clear on match start and internal clock.
- In MC mode with Clear on Match End: Timebase generation with clear on match end.
- In OPWFM mode: Timebase generation with clear on match start and internal clock. The internal counter clears at the start of the match signal, skips the next prescaled clock edge, and then increments in the subsequent prescaled clock edge.

60.8.3 Timebase with the fastest prescaler ratio

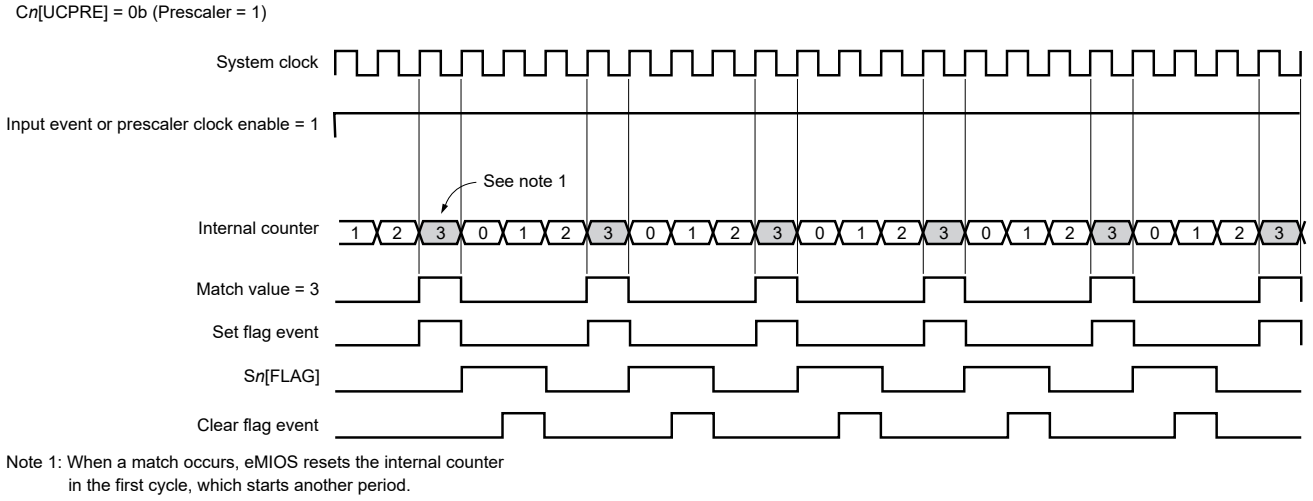
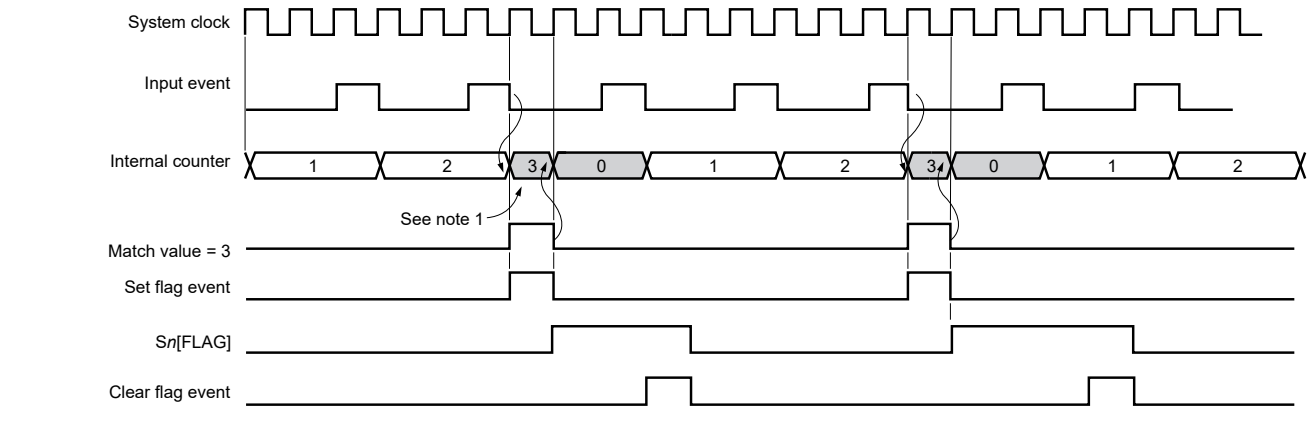


Figure 340. Timebase with fastest prescaler ratio

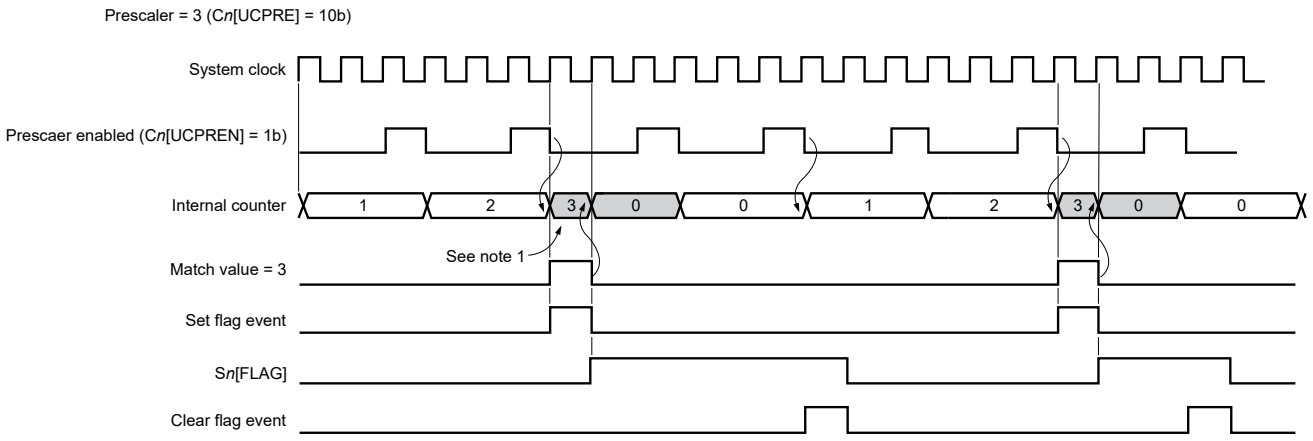
60.8.4 Timebase generation with clear on match start and external clock



Note 1: When a match occurs, eMIOS resets the internal counter in the first clock cycle.
The counter starts counting at the next edge of the prescaler clock enable.

Figure 341. Timebase generation with clear on match start and external clock

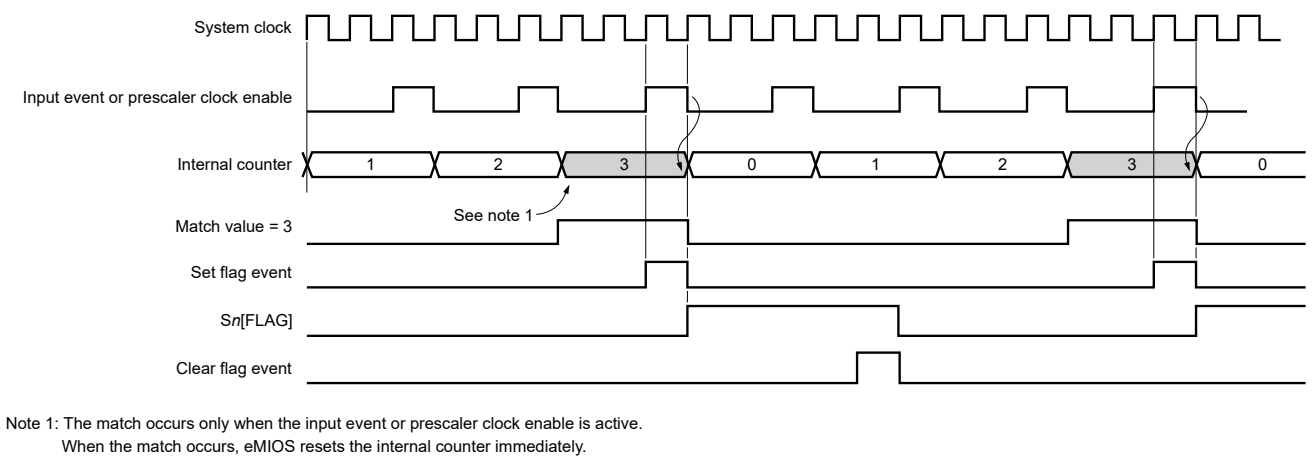
60.8.5 Timebase generation with clear on match start and internal clock



Note 1: When a match occurs, eMIOS clears the internal counter in the first clock cycle.
The counter starts counting again only after the second edge of the prescaled clock.

Figure 342. Timebase generation with clear on match start and internal clock

60.8.6 Timebase generation with clear on match end



Note 1: The match occurs only when the input event or prescaler clock enable is active.
When the match occurs, eMIOS resets the internal counter immediately.

Figure 343. Timebase generation with clear on match end

60.8.7 Coherent accesses

Reading A*n*[A] again in the same period as the last read of B*n*[B] results in incoherent results if the last read of B*n*[B] occurred after a disabled BS2→BS1 transfer.

60.8.8 Initializing channels and modes

Perform the following basic steps for output mode startup. This procedure assumes the channels are initially in GPIO mode:

1. Disable the GPC (write 0 to MCR[GPREN]).
2. For each timebase UC:
    a. Disable the CP (write 0 to C*n*[UCPREN]).
    b. Write an initial value to the internal counter (CNT*n*[C]).
    c. Write initial values to A*n*[A] and B*n*[B].
    d. Configure the UC for MC or MCB Up mode (write the appropriate value to C*n*[MODE]).
    e. Select the CP ratio (C*n*[UCPRE]).
    f. Enable the CP (write 1 to C*n*[UCPREN]).
3. For each output UC:
    a. Disable the CP (write 0 to C*n*[UCPREN]).
    b. Write initial values to A*n*[A] and B*n*[B].
    c. Select the timebase input (C*n*[BSL]).
    d. Specify the output mode (write the appropriate value to C*n*[MODE]).
    e. Select the same CP ratio as the timebase UC (C*n*[UCPRE]).
    f. Enable the CP (write 1 to C*n*[UCPREN]).
4. Enable the GPC (write 1 to MCR[GPREN]).
5. Enable GBTE (write 1 to MCR[GBTE]).

The timebase channel and the output channel may be the same for some applications such as in OPWFM or OPWFMB mode, or whenever the output channel drives the timebase itself.

You may configure flags at any time.

60.9 Memory maps and registers

60.9.1 Overall address map organization

All address locations not explicitly mentioned in this table are reserved. When you access an absent register, absent channel, or reserved address space, the transfer terminates with an error.

Table 343. Overall address map organization

Base address (hex)	Description
0	Module Configuration (MCR)
4	Global Flag (GFLAG)
8	Output Update Disable (OUDIS)
C	Disable Channel (UCDIS)
20	Channels 0–7
120	Channels 8–15
220	Channels 16–23

60.9.2 UC memory overview

Table 344. UC memory overview

Offset from UC *n* base address (hex)	Description
0	UC A n (A0 - A23)
4	UC B n (B0 - B23)
8	UC Counter n (CNT0 - CNT23)
C	UC Control n (C0 - C23)
10	UC Status n (S0 - S23)
14	Alternate Address n (ALTA0 - ALTA23)
18	UC Control 2 n (C2_0 - C2_23)
1C–1F	Reserved

60.9.3 AS1, AS2, BS1, and BS2 shadow registers

When you use eMIOS UC modes, you rely heavily on four shadow registers:

- AS1
- AS2
- BS1
- BS2

You use A*n*[A], B*n*[B], and ALTA*n*[ALTA] to access these shadow registers:

- Each instance of A*n*[A] has a separate associated instance of AS1 and AS2. In some UC modes, you also use ALTA*n*[ALTA] to access AS2.
- Each instance of B*n*[B] has a separate associated instance of BS1 and BS2.

See Relationship between A*n*, B*n*, ALTA*n*, and shadow registers.

Each UC mode uses the AS1-AS2 and BS1-BS2 shadow register pairs for various match and capture functions, as described in Unified channels (UC).

A reset clears all shadow registers.

Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA] describes how you write to and read from the shadow registers for all operation modes. Values not listed are reserved. For more information, see Unified channels (UC).

60.9.4 Relationship between A*n*, B*n*, ALTA*n*, and shadow registers

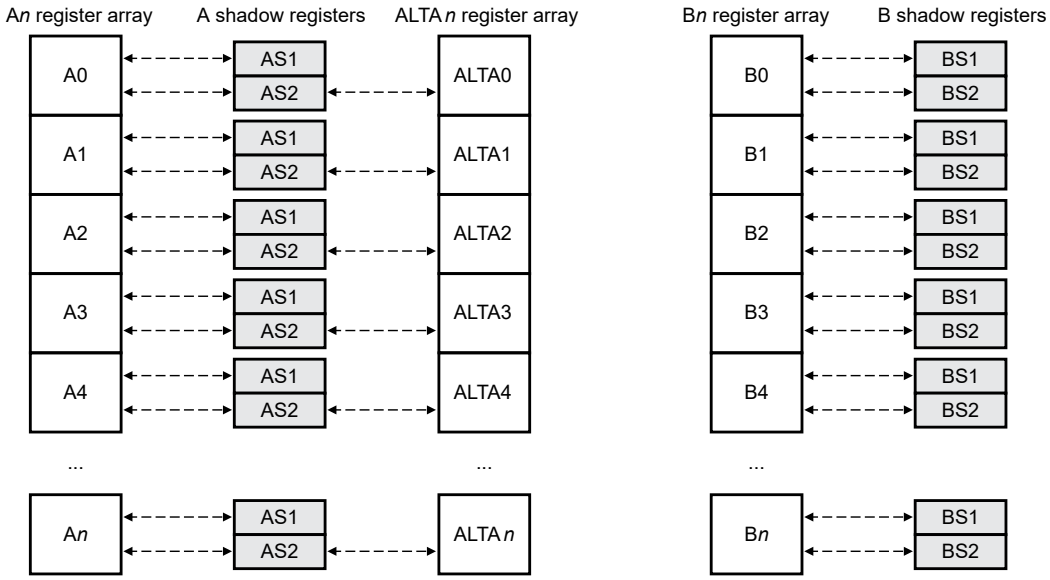


Figure 344. Relationship between A*n*, B*n*, ALTA*n*, and shadow registers

60.9.5 Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA]

This table describes how you write to and read from the following registers for all operation modes:

- UC A n (A0 - A23)
- UC B n (B0 - B23)
- Alternate Address n (ALTA0 - ALTA23)

Values not listed are reserved. For more information, see Unified channels (UC).

Table 345. Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA]

Operation mode	Register access					
	A*n*		B*n*		ALTA*n*	
	Write	Read	Write	Read	Write	Read
GPIO	AS1 and AS2[1]	AS1	BS1 and BS2[2]	BS1	AS2	AS2

Table continues on the next page...

Table 345. Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA] (continued)

Operation mode	Register access					
	A*n*		B*n*		ALTA*n*	
	Write	Read	Write	Read	Write	Read
SAIC[3]	—	AS2	BS2	BS2	—	—
SAOC[3]	AS2	AS1	BS2	BS2	—	—
IPWM	—	AS2	—	BS1	—	—
IPM	—	AS2	—	BS1	—	—
DAOC	AS2	AS1	BS2	BS1	—	—
PEA	AS1	AS2	—	BS1	—	—
PEC[3]	AS1	AS1	BS1	BS1	—	AS2
WPTA	AS1	AS1	BS1	BS1	—	AS2
MC[3]	AS2	AS1	BS2	BS2	—	—
OPWFM	AS2	AS1	BS2	BS1	—	—
OPWMC	AS2	AS1	BS2	BS1	—	—
OPWM	AS2	AS1	BS2	BS1	—	—
OPWMT	AS1	AS1	BS2	BS1	AS2	AS2
MCB[3]	AS2	AS1	BS2	BS1	—	—
OPWFMB	AS2	AS1	BS2	BS1	—	—
OPWMCB	AS2	AS1	BS2	BS1	—	—
OPWMB	AS2	AS1	BS2	BS1	—	—

1. This operation writes the same value to AS1 and AS2.
2. This operation writes the same value to BS1 and BS2.
3. These modes do not require UC B n (B0 - B23), but you can still use it to access BS2.

60.9.6 eMIOS register descriptions

All control registers are 32 bits wide. Data fields are 16 bits wide. There are 24 UCs, as mapped in Counter buses, channels, and timebase sources.

60.9.6.1 eMIOS memory map

eMIOS_0 base address: 4008_8000h

eMIOS_1 base address: 4008_C000h

eMIOS_2 base address: 4009_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration (MCR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4h	Global Flag (GFLAG)	32	R	0000_0000h
8h	Output Update Disable (OUDIS)	32	RW	0000_0000h
Ch	Disable Channel (UCDIS)	32	RW	0000_0000h
20h	UC A 0 (A0)	32	RW	0000_0000h
24h	UC B 0 (B0)	32	RW	0000_0000h
28h	UC Counter 0 (CNT0)	32	RW	0000_0000h
2Ch	UC Control 0 (C0)	32	RW	0000_0000h
30h	UC Status 0 (S0)	32	RW	See section
34h	Alternate Address 0 (ALTA0)	32	RW	0000_0000h
38h	UC Control 2 0 (C2_0)	32	RW	0000_0000h
40h	UC A 1 (A1)	32	RW	0000_0000h
44h	UC B 1 (B1)	32	RW	0000_0000h
48h	UC Counter 1 (CNT1)	32	RW	0000_0000h
4Ch	UC Control 1 (C1)	32	RW	0000_0000h
50h	UC Status 1 (S1)	32	RW	See section
54h	Alternate Address 1 (ALTA1)	32	RW	0000_0000h
58h	UC Control 2 1 (C2_1)	32	RW	0000_0000h
60h	UC A 2 (A2)	32	RW	0000_0000h
64h	UC B 2 (B2)	32	RW	0000_0000h
68h	UC Counter 2 (CNT2)	32	RW	0000_0000h
6Ch	UC Control 2 (C2)	32	RW	0000_0000h
70h	UC Status 2 (S2)	32	RW	See section
74h	Alternate Address 2 (ALTA2)	32	RW	0000_0000h
78h	UC Control 2 2 (C2_2)	32	RW	0000_0000h
80h	UC A 3 (A3)	32	RW	0000_0000h
84h	UC B 3 (B3)	32	RW	0000_0000h
88h	UC Counter 3 (CNT3)	32	RW	0000_0000h
8Ch	UC Control 3 (C3)	32	RW	0000_0000h
90h	UC Status 3 (S3)	32	RW	See section
94h	Alternate Address 3 (ALTA3)	32	RW	0000_0000h
98h	UC Control 2 3 (C2_3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A0h	UC A 4 (A4)	32	RW	0000_0000h
A4h	UC B 4 (B4)	32	RW	0000_0000h
A8h	UC Counter 4 (CNT4)	32	RW	0000_0000h
ACh	UC Control 4 (C4)	32	RW	0000_0000h
B0h	UC Status 4 (S4)	32	RW	See section
B4h	Alternate Address 4 (ALTA4)	32	RW	0000_0000h
B8h	UC Control 2 4 (C2_4)	32	RW	0000_0000h
C0h	UC A 5 (A5)	32	RW	0000_0000h
C4h	UC B 5 (B5)	32	RW	0000_0000h
C8h	UC Counter 5 (CNT5)	32	RW	0000_0000h
CCh	UC Control 5 (C5)	32	RW	0000_0000h
D0h	UC Status 5 (S5)	32	RW	See section
D4h	Alternate Address 5 (ALTA5)	32	RW	0000_0000h
D8h	UC Control 2 5 (C2_5)	32	RW	0000_0000h
E0h	UC A 6 (A6)	32	RW	0000_0000h
E4h	UC B 6 (B6)	32	RW	0000_0000h
E8h	UC Counter 6 (CNT6)	32	RW	0000_0000h
ECh	UC Control 6 (C6)	32	RW	0000_0000h
F0h	UC Status 6 (S6)	32	RW	See section
F4h	Alternate Address 6 (ALTA6)	32	RW	0000_0000h
F8h	UC Control 2 6 (C2_6)	32	RW	0000_0000h
100h	UC A 7 (A7)	32	RW	0000_0000h
104h	UC B 7 (B7)	32	RW	0000_0000h
108h	UC Counter 7 (CNT7)	32	RW	0000_0000h
10Ch	UC Control 7 (C7)	32	RW	0000_0000h
110h	UC Status 7 (S7)	32	RW	See section
114h	Alternate Address 7 (ALTA7)	32	RW	0000_0000h
118h	UC Control 2 7 (C2_7)	32	RW	0000_0000h
120h	UC A 8 (A8)	32	RW	0000_0000h
124h	UC B 8 (B8)	32	RW	0000_0000h
128h	UC Counter 8 (CNT8)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
12Ch	UC Control 8 (C8)	32	RW	0000_0000h
130h	UC Status 8 (S8)	32	RW	See section
134h	Alternate Address 8 (ALTA8)	32	RW	0000_0000h
138h	UC Control 2 8 (C2_8)	32	RW	0000_0000h
140h	UC A 9 (A9)	32	RW	0000_0000h
144h	UC B 9 (B9)	32	RW	0000_0000h
14Ch	UC Control 9 (C9)	32	RW	0000_0000h
150h	UC Status 9 (S9)	32	RW	See section
154h	Alternate Address 9 (ALTA9)	32	RW	0000_0000h
158h	UC Control 2 9 (C2_9)	32	RW	0000_0000h
160h	UC A 10 (A10)	32	RW	0000_0000h
164h	UC B 10 (B10)	32	RW	0000_0000h
16Ch	UC Control 10 (C10)	32	RW	0000_0000h
170h	UC Status 10 (S10)	32	RW	See section
174h	Alternate Address 10 (ALTA10)	32	RW	0000_0000h
178h	UC Control 2 10 (C2_10)	32	RW	0000_0000h
180h	UC A 11 (A11)	32	RW	0000_0000h
184h	UC B 11 (B11)	32	RW	0000_0000h
18Ch	UC Control 11 (C11)	32	RW	0000_0000h
190h	UC Status 11 (S11)	32	RW	See section
194h	Alternate Address 11 (ALTA11)	32	RW	0000_0000h
198h	UC Control 2 11 (C2_11)	32	RW	0000_0000h
1A0h	UC A 12 (A12)	32	RW	0000_0000h
1A4h	UC B 12 (B12)	32	RW	0000_0000h
1ACh	UC Control 12 (C12)	32	RW	0000_0000h
1B0h	UC Status 12 (S12)	32	RW	See section
1B4h	Alternate Address 12 (ALTA12)	32	RW	0000_0000h
1B8h	UC Control 2 12 (C2_12)	32	RW	0000_0000h
1C0h	UC A 13 (A13)	32	RW	0000_0000h
1C4h	UC B 13 (B13)	32	RW	0000_0000h
1CCh	UC Control 13 (C13)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1D0h	UC Status 13 (S13)	32	RW	See section
1D4h	Alternate Address 13 (ALTA13)	32	RW	0000_0000h
1D8h	UC Control 2 13 (C2_13)	32	RW	0000_0000h
1E0h	UC A 14 (A14)	32	RW	0000_0000h
1E4h	UC B 14 (B14)	32	RW	0000_0000h
1ECh	UC Control 14 (C14)	32	RW	0000_0000h
1F0h	UC Status 14 (S14)	32	RW	See section
1F4h	Alternate Address 14 (ALTA14)	32	RW	0000_0000h
1F8h	UC Control 2 14 (C2_14)	32	RW	0000_0000h
200h	UC A 15 (A15)	32	RW	0000_0000h
204h	UC B 15 (B15)	32	RW	0000_0000h
20Ch	UC Control 15 (C15)	32	RW	0000_0000h
210h	UC Status 15 (S15)	32	RW	See section
214h	Alternate Address 15 (ALTA15)	32	RW	0000_0000h
218h	UC Control 2 15 (C2_15)	32	RW	0000_0000h
220h	UC A 16 (A16)	32	RW	0000_0000h
224h	UC B 16 (B16)	32	RW	0000_0000h
228h	UC Counter 16 (CNT16)	32	RW	0000_0000h
22Ch	UC Control 16 (C16)	32	RW	0000_0000h
230h	UC Status 16 (S16)	32	RW	See section
234h	Alternate Address 16 (ALTA16)	32	RW	0000_0000h
238h	UC Control 2 16 (C2_16)	32	RW	0000_0000h
240h	UC A 17 (A17)	32	RW	0000_0000h
244h	UC B 17 (B17)	32	RW	0000_0000h
24Ch	UC Control 17 (C17)	32	RW	0000_0000h
250h	UC Status 17 (S17)	32	RW	See section
254h	Alternate Address 17 (ALTA17)	32	RW	0000_0000h
258h	UC Control 2 17 (C2_17)	32	RW	0000_0000h
260h	UC A 18 (A18)	32	RW	0000_0000h
264h	UC B 18 (B18)	32	RW	0000_0000h
26Ch	UC Control 18 (C18)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
270h	UC Status 18 (S18)	32	RW	See section
274h	Alternate Address 18 (ALTA18)	32	RW	0000_0000h
278h	UC Control 2 18 (C2_18)	32	RW	0000_0000h
280h	UC A 19 (A19)	32	RW	0000_0000h
284h	UC B 19 (B19)	32	RW	0000_0000h
28Ch	UC Control 19 (C19)	32	RW	0000_0000h
290h	UC Status 19 (S19)	32	RW	See section
294h	Alternate Address 19 (ALTA19)	32	RW	0000_0000h
298h	UC Control 2 19 (C2_19)	32	RW	0000_0000h
2A0h	UC A 20 (A20)	32	RW	0000_0000h
2A4h	UC B 20 (B20)	32	RW	0000_0000h
2ACh	UC Control 20 (C20)	32	RW	0000_0000h
2B0h	UC Status 20 (S20)	32	RW	See section
2B4h	Alternate Address 20 (ALTA20)	32	RW	0000_0000h
2B8h	UC Control 2 20 (C2_20)	32	RW	0000_0000h
2C0h	UC A 21 (A21)	32	RW	0000_0000h
2C4h	UC B 21 (B21)	32	RW	0000_0000h
2CCh	UC Control 21 (C21)	32	RW	0000_0000h
2D0h	UC Status 21 (S21)	32	RW	See section
2D4h	Alternate Address 21 (ALTA21)	32	RW	0000_0000h
2D8h	UC Control 2 21 (C2_21)	32	RW	0000_0000h
2E0h	UC A 22 (A22)	32	RW	0000_0000h
2E4h	UC B 22 (B22)	32	RW	0000_0000h
2E8h	UC Counter 22 (CNT22)	32	RW	0000_0000h
2ECh	UC Control 22 (C22)	32	RW	0000_0000h
2F0h	UC Status 22 (S22)	32	RW	See section
2F4h	Alternate Address 22 (ALTA22)	32	RW	0000_0000h
2F8h	UC Control 2 22 (C2_22)	32	RW	0000_0000h
300h	UC A 23 (A23)	32	RW	0000_0000h
304h	UC B 23 (B23)	32	RW	0000_0000h
308h	UC Counter 23 (CNT23)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
30Ch	UC Control 23 (C23)	32	RW	0000_0000h
310h	UC Status 23 (S23)	32	RW	See section
314h	Alternate Address 23 (ALTA23)	32	RW	0000_0000h
318h	UC Control 2 23 (C2_23)	32	RW	0000_0000h

60.9.6.2 Module Configuration (MCR)

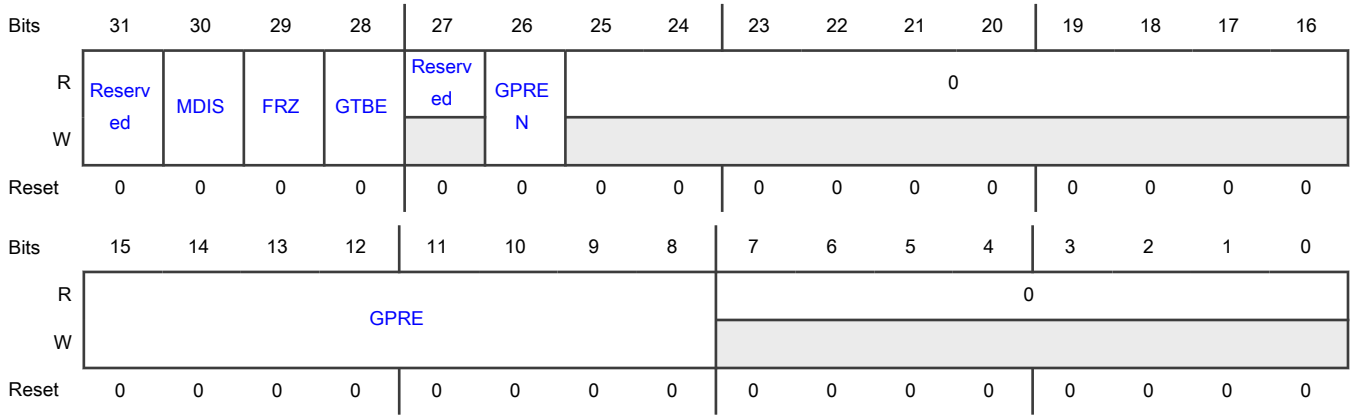
Offset

Register	Offset
MCR	0h

Function

Configures eMIOS operation.

Diagram



Fields

Field	Function
31 —	Reserved Write only the reset value to this field.
30 MDIS	Module Disable Disables eMIOS by stopping the clock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When you disable eMIOS, its clock stops, it consumes less power, and you can access only the following registers: • Module Configuration (MCR) • Output Update Disable (OUDIS) • Disable Channel (UCDIS) 0b - Enable 1b - Disable
29 FRZ	Freeze Prepares eMIOS to enter Freeze state (see Freeze and chip Debug mode). eMIOS does not enter Freeze state until both of the following conditions occur: • The chip asserts Debug mode. • You enable Freeze for each UC (write 1 to C*n*[FREN]). A channel remains frozen until one or more of the following occur: • You force eMIOS to exit the Freeze state (write 0 to FRZ). • The chip deasserts Debug mode. • You disable Freeze for that UC (write 0 to C*n*[FREN]). 0b - Exit Freeze state 1b - Enter Freeze state
28 GTBE	Global Timebase Enable Asserts the GTBE_OUT signal. If GTBE_OUT is connected to the GTBE_IN input of one or more eMIOS instances, asserting the signal turns on the internal counters of those eMIOS instances simultaneously. **NOTE** The GTBE_IN input enables (asserted) or disables (deasserted) the internal counters. 0b - Deassert GTBE_OUT 1b - Assert GTBE_OUT
27 —	Reserved
26 GPREN	Global Prescaler Enable Enables the global prescaler counter. Disabling the global prescaler clears the prescaler counter and stops the prescaler clock. 0b - Disable 1b - Enable
25-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-8 GPRE	Global Prescaler Specifies the global clock prescaler. Write the desired clock divide ratio minus 1. For example, for a divide ratio of 8, write 111b. The prescaler can range from 1 (0b) to 256 (11111111b).
7-0 —	Reserved

60.9.6.3 Global Flag (GFLAG)

Offset

Register	Offset
GFLAG	4h

Function

Groups the flag fields from all channels into one register to improve interrupt handling.

For UCs, each field mirrors the corresponding channel flag (S*n*[FLAG]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	F23	F22	F21	F20	F19	F18	F17	F16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 F23	Mirror of UC 23 FLAG
22 F22	Mirror of UC 22 FLAG
21 F21	Mirror of UC 21 FLAG
20 F20	Mirror of UC 20 FLAG
19 F19	Mirror of UC 19 FLAG
18 F18	Mirror of UC 18 FLAG
17 F17	Mirror of UC 17 FLAG
16 F16	Mirror of UC 16 FLAG

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 F15	Mirror of UC 15 FLAG
14 F14	Mirror of UC 14 FLAG
13 F13	Mirror of UC 13 FLAG
12 F12	Mirror of UC 12 FLAG
11 F11	Mirror of UC 11 FLAG
10 F10	Mirror of UC 10 FLAG
9 F9	Mirror of UC 9 FLAG
8 F8	Mirror of UC 8 FLAG
7 F7	Mirror of UC 7 FLAG
6 F6	Mirror of UC 6 FLAG
5 F5	Mirror of UC 5 FLAG
4 F4	Mirror of UC 4 FLAG
3 F3	Mirror of UC 3 FLAG
2 F2	Mirror of UC 2 FLAG
1	Mirror of UC 1 FLAG

Table continues on the next page...

Table continued from the previous page...

Field	Function
F1	
0 F0	Mirror of UC 0 FLAG

60.9.6.4 Output Update Disable (OUDIS)

Offset

Register	Offset
OUDIS	8h

Function

Disables transfers from AS2 to AS1 and BS2 to BS1 on a per-channel basis. This applies to any channel operating in MC, MCB, or any output mode that writes to AS2 and BS2.

When a field is 0, transfers on that channel occur immediately or in the next period, depending on the operation mode. Unless that mode's description states otherwise, the transfer occurs immediately.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	OU23	OU22	OU21	OU20	OU19	OU18	OU17	OU16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OU15	OU14	OU13	OU12	OU11	OU10	OU9	OU8	OU7	OU6	OU5	OU4	OU3	OU2	OU1	OU0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 OU23	Channel 23 Output Update Disable 0b - Enable 1b - Disable
22 OU22	Channel 22 Output Update Disable 0b - Enable 1b - Disable
21 OU21	Channel 21 Output Update Disable 0b - Enable 1b - Disable
20 OU20	Channel 20 Output Update Disable 0b - Enable 1b - Disable
19 OU19	Channel 19 Output Update Disable 0b - Enable 1b - Disable
18 OU18	Channel 18 Output Update Disable 0b - Enable 1b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 OU17	Channel 17 Output Update Disable 0b - Enable 1b - Disable
16 OU16	Channel 16 Output Update Disable 0b - Enable 1b - Disable
15 OU15	Channel 15 Output Update Disable 0b - Enable 1b - Disable
14 OU14	Channel 14 Output Update Disable 0b - Enable 1b - Disable
13 OU13	Channel 13 Output Update Disable 0b - Enable 1b - Disable
12 OU12	Channel 12 Output Update Disable 0b - Enable 1b - Disable
11 OU11	Channel 11 Output Update Disable 0b - Enable 1b - Disable
10 OU10	Channel 10 Output Update Disable 0b - Enable 1b - Disable
9 OU9	Channel 9 Output Update Disable 0b - Enable 1b - Disable
8 OU8	Channel 8 Output Update Disable 0b - Enable 1b - Disable
7 OU7	Channel 7 Output Update Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enable 1b - Disable
6 OU6	Channel 6 Output Update Disable 0b - Enable 1b - Disable
5 OU5	Channel 5 Output Update Disable 0b - Enable 1b - Disable
4 OU4	Channel 4 Output Update Disable 0b - Enable 1b - Disable
3 OU3	Channel 3 Output Update Disable 0b - Enable 1b - Disable
2 OU2	Channel 2 Output Update Disable 0b - Enable 1b - Disable
1 OU1	Channel 1 Output Update Disable 0b - Enable 1b - Disable
0 OU0	Channel 0 Output Update Disable 0b - Enable 1b - Disable

60.9.6.5 Disable Channel (UCDIS)

Offset

Register	Offset
UCDIS	Ch

Function

Allows you to disable a UC by stopping its clock. Each field controls the clock for the corresponding UC.

Diagram

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

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

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 UCDIS23	Disable UC 23 0b - Enable 1b - Disable
22	Disable UC 22

Table continues on the next page...

Table continued from the previous page...

Field	Function
UCDIS22	0b - Enable 1b - Disable
21 UCDIS21	Disable UC 21 0b - Enable 1b - Disable
20 UCDIS20	Disable UC 20 0b - Enable 1b - Disable
19 UCDIS19	Disable UC 19 0b - Enable 1b - Disable
18 UCDIS18	Disable UC 18 0b - Enable 1b - Disable
17 UCDIS17	Disable UC 17 0b - Enable 1b - Disable
16 UCDIS16	Disable UC 16 0b - Enable 1b - Disable
15 UCDIS15	Disable UC 15 0b - Enable 1b - Disable
14 UCDIS14	Disable UC 14 0b - Enable 1b - Disable
13 UCDIS13	Disable UC 13 0b - Enable 1b - Disable
12 UCDIS12	Disable UC 12 0b - Enable 1b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 UCDIS11	Disable UC 11 0b - Enable 1b - Disable
10 UCDIS10	Disable UC 10 0b - Enable 1b - Disable
9 UCDIS9	Disable UC 9 0b - Enable 1b - Disable
8 UCDIS8	Disable UC 8 0b - Enable 1b - Disable
7 UCDIS7	Disable UC 7 0b - Enable 1b - Disable
6 UCDIS6	Disable UC 6 0b - Enable 1b - Disable
5 UCDIS5	Disable UC 5 0b - Enable 1b - Disable
4 UCDIS4	Disable UC 4 0b - Enable 1b - Disable
3 UCDIS3	Disable UC 3 0b - Enable 1b - Disable
2 UCDIS2	Disable UC 2 0b - Enable 1b - Disable
1 UCDIS1	Disable UC 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enable 1b - Disable
0 UCDIS0	Disable UC 0 0b - Enable 1b - Disable

60.9.6.6 UC A n (A0 - A23)

Offset

For n = 0 to 23:

Register	Offset
An	20h + (n × 20h)

Function

Accesses the AS1 and AS2 shadow registers for each UC, depending on the selected UC mode. See AS1, AS2, BS1, and BS2 shadow registers.

For information about how you write to and read from A*n*, B*n*, and ALTA*n* for all UC modes, see Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA].

Diagram

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

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

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 A	A See the register description.

60.9.6.7 UC B n (B0 - B23)

Offset

For n = 0 to 23:

Register	Offset
Bn	24h + (n × 20h)

Function

Accesses the BS1 and BS2 shadow registers for each UC, depending on the selected UC mode. See AS1, AS2, BS1, and BS2 shadow registers.

For information about how you write to and read from A*n*, B*n*, and ALTA*n* for all UC modes, see Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA].

Diagram

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

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

Fields

Field	Function
31-16 —	Reserved
15-0 B	B See the register description.

60.9.6.8 UC Counter n (CNT0 - CNT23)

Offset

Register	Offset
CNT0	28h
CNT1	48h
CNT2	68h
CNT3	88h
CNT4	A8h
CNT5	C8h
CNT6	E8h
CNT7	108h
CNT8	128h
CNT16	228h
CNT22	2E8h
CNT23	308h

Function

Indicates the value of the UC internal counter.

When eMIOS is in GPIO mode or the UC is frozen, this register is read-write. For all other modes, this register is read-only. When entering some UC modes, eMIOS automatically clears this register. See Unified channels (UC) for details.

The following modes use the UC counter:

- Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode
- Center Aligned Output PWM with Dead Time Insertion Buffered (OPWMCB) mode
- Pulse Edge Counting (PEC) mode
- Modulus Counter (MC) mode
- Modulus Counter Buffered (MCB) mode

All other modes not in this list do not use the internal counter.

NOTE

Each module instance supports a different number of registers.

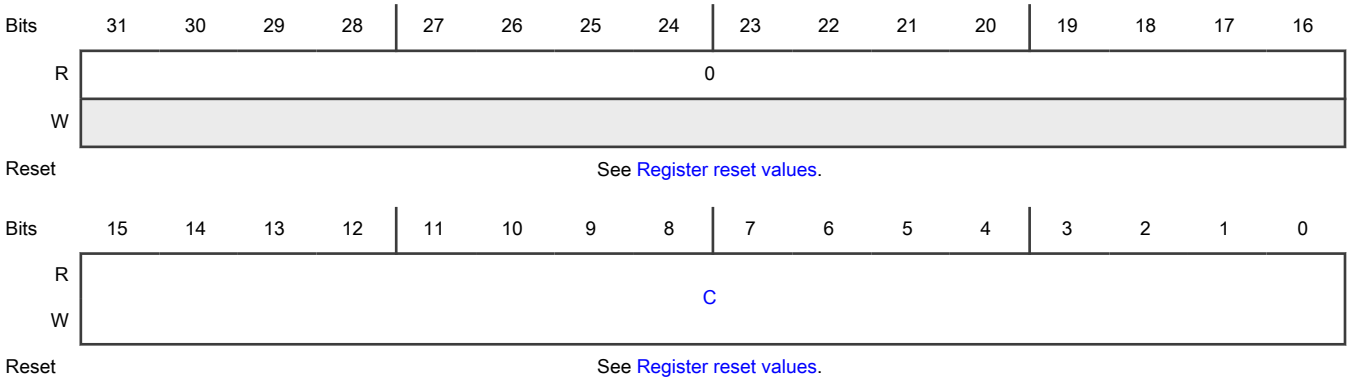
Instance	Register supported	Register not supported
eMIOS_0	CNT0–CNT8 CNT16 CNT22–CNT23	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
eMIOS_1	CNT0 CNT8 CNT16 CNT22–CNT23	CNT1–CNT7
eMIOS_2	CNT0 CNT8 CNT16 CNT22–CNT23	CNT1–CNT7

Diagram



Register reset values

Register	Reset value
CNT0	eMIOS_0–eMIOS_2: 0000_0000h
CNT1–CNT7	0000_0000h
CNT8–CNT23	eMIOS_0–eMIOS_2: 0000_0000h
CNT9–CNT15	Register not supported
CNT17–CNT21	Register not supported

Fields

Field	Function
31-16 —	Reserved
15-0 C	Internal Counter Value Indicates the value of the internal counter.

60.9.6.9 UC Control n (C0 - C23)

Offset

For n = 0 to 23:

Register	Offset
Cn	2Ch + (n × 20h)

Function

Controls UC operation.

Table 346. MODE field values

UC mode	MODE (binary)
GPIO (input)	000_0000
GPIO (output)	000_0001
SAIC	000_0010
SAIC with edge capturing	100_0010
SAOC	000_0011
IPWM	000_0100
IPM	000_0101
DAOC (with FLAG = 1 on B match)	000_0110
DAOC (with FLAG = 1 on A and B match)	000_0111
PEC (continuous)	000_1010
PEC (single shot)	000_1011
Reserved	000_1111
MC (up counter with clear on match start)	001_000p[1]
MC (up counter with clear on match end)	001_001p[1]
MC (up/down counter)	001_01pp[1]
Reserved	010_0100–010_0101
OPWMT	010_0110
Reserved	010_0111–100_1111
MCB (up counter)	101_000p[1]
Reserved	101_001p
MCB (up or down counter)	101_01pp[1]
OPWFMB	101_10p0[1]
Reserved	101_10p1
OPWMCB (with trailing edge dead time)	101_11p0[1]

Table continues on the next page...

Table 346. MODE field values (continued)

UC mode	MODE (binary)
OPWMCB (with leading edge dead time)	101_11p1[1]
OPWMB	110_00p0[1]
Reserved	110_0001–111_1111

1. p = Adjust parameters for the mode of operation. See Unified channels (UC) for details.

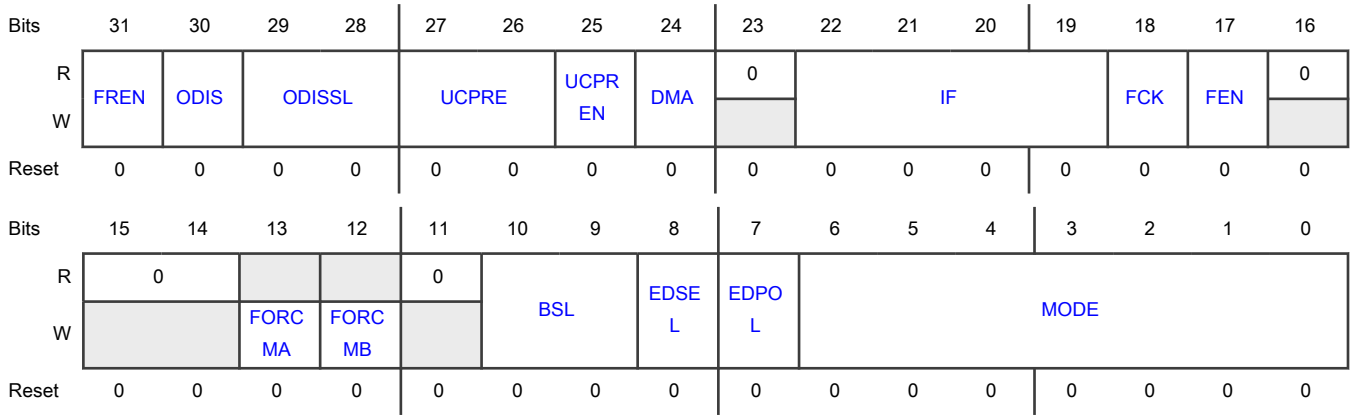
Table 347. Edge Polarity (EDPOL) settings

For these modes:	EDPOL does this:	and EDPOL values mean:
Input	Selects which edge triggers the internal counter, an input capture, or the input capture flag. If EDPOL does not appear in the UC mode description, it has no effect on UC operation.	0b - Trigger on a falling edge 1b - Trigger on a rising edge
Output	Selects the logic level on the output pin.	0b - A match on comparator A deasserts the output flip-flop, while a match on comparator B sets it 1b - A match on comparator A asserts the output flip-flop, while a match on comparator B clears it

Table 348. Edge Select (EDSEL) settings

For these modes:	EDSEL does this:	and EDSEL values mean:
Input	Selects whether the internal counter is triggered by both edges of a pulse or by a single edge as defined by EDPOL. If EDSEL does not appear in the UC mode description, it has no effect on UC operation.	0b - Single-edge triggering on edge specified in EDPOL 1b - Both-edges triggering
GPIO in	Selects whether an edge asserts the input capture flag (S*n*[FLAG]).	0b - Sets S*n*[FLAG] on the edge specified in EDPOL 1b - Does not set S*n*[FLAG]
SAOC	Selects the behavior of the output flip-flop at each match.	0b - Drive the EDPOL value on the output flip-flop 1b - Toggle output flip-flop

Diagram



Fields

Field	Function
31 FREN	Freeze Enable Enables putting the UC in Freeze state. See Freeze and chip Debug mode. 0b - Disable 1b - Enable
30 ODIS	Output Disable Disables the output pin when running any of the output modes except GPIO. When you assert ODIS: • If the selected Output Disable Input signal asserts, the output pin reflects: — For OPWFMB and OPWMB modes, EDPOL (see Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode and Output PWM Buffered (OPWMB) mode). — For all other modes, the complement of EDPOL, but the UC continues to operate normally—it continues to generate flags and matches. • If the selected Output Disable Input signal deasserts, the output pin operates normally. 0b - Enables output pin (the output pin operates normally) 1b - Disables output pin
29-28 ODISSL	Output Disable Select Selects one of four output disable input signals. 00b - 0 01b - 1 10b - 2 11b - 3
27-26	Prescaler Selects the clock divider value for the UC internal prescaler.

Table continues on the next page...

Table continued from the previous page...

Field	Function
UCPRE	**NOTE** The two least-significant bits of C2_*n*[UCEXTPRE] mirror UCPRE. Any write to UCPRE also affects UCEXTPRE. 00b - 1 01b - 2 10b - 3 11b - 4
25 UCPREN	Prescaler Enable Enables the prescaler counter. 0b - Disable (no clock) 1b - Enable
24 DMA	Direct Memory Access Selects whether the input capture flag (S*n*[FLAG]) or overrun (S*n*[OVR]) triggers an interrupt or DMA request. 0b - Interrupt request 1b - DMA request
23 —	Reserved
22-19 IF	Input Filter Selects the minimum input pulse width, in filter clock cycles, that can pass through the input filter. This field does not apply to output modes. The filter latency—the difference in time between the input and the response—is three clock edges. 0000b - Bypassed; the input signal is synchronized before arriving at the digital filter 0001b - 2 cycles 0010b - 4 cycles 0100b - 8 cycles 1000b - 16 cycles All other values are reserved.
18 FCK	Filter Clock Select Selects the clock source for the programmable input filter. 0b - Prescaled clock 1b - eMIOS module clock

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 FEN	Flag Enable Enables the input capture flag (S*n*[FLAG]) to generate an interrupt request or a DMA request, as selected in DMA. 0b - Disable 1b - Enable
16-14 —	Reserved
13 FORCMA	Force Match A For output modes, asserting Force Match A causes the comparator to report a successful comparison on comparator A regardless of whether a comparison actually occurred. It does not set the input capture flag (S*n*[FLAG]). This field always reads 0. This bit is meaningful for every output operation mode that uses comparator A. Otherwise it has no effect. 0b - No effect 1b - Force a match at comparator A
12 FORCMB	Force Match B For output modes, asserting Force Match B causes the comparator to report a successful comparison on comparator B regardless of whether a comparison actually occurred. It does not set the input capture flag (S*n*[FLAG] does not become 1). This field always reads 0. This field applies only if the UC is in one of the following output modes. • Single Action Output Capture (SAOC) mode • Double Action Output Compare (DAOC) mode • Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode • Center Aligned Output PWM with Dead Time Insertion Buffered (OPWMCB) mode • Output PWM Buffered (OPWMB) mode • Output PWM with Trigger (OPWMT) mode This field has no effect in all other modes. 0b - No effect 1b - Force a match at comparator B
11 —	Reserved
10-9 BSL	Bus Select Selects one of the counter buses or the internal counter for the UC. Not all chips support all counter buses. See the chip-specific eMIOS information for details. 00b -

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Channels 0-22: counter bus A • Channel 23: reserved 01b - • Channels 1-7: counter bus B • Channels 9-15: counter bus C • Channels 17-23: counter bus D • Channels 25-31: counter bus E • Channels 0, 8, 16, 24: reserved 10b - • Channels 0-21, 23: counter bus F • Channel 22: reserved 11b - Internal counter for all channels
8 EDSEL	Edge Selection Controls several functions as shown in Table 348.
7 EDPOL	Edge Polarity Controls several functions as shown in Table 347.
6-0 MODE	Mode Selection Selects the UC mode as shown in Table 346. **NOTE** If you write a reserved value to this field, the results are unpredictable. When you change the value of MODE, you must first enter GPIO mode to reset the UC's internal functions. Failure to do this can result in invalid and unexpected output compare or input capture results or incorrectly set input capture flags (S*n*[FLAG]).

60.9.6.10 UC Status n (S0 - S23)

Offset

For n = 0 to 23:

Register	Offset
Sn	30h + (n × 20h)

Function

Indicates the status of the UC input-output signals and the overflow condition of the internal counter.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OVR	0														
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OVFL	0												UCIN	UCOUT	FLAG
W	W1C															W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	u[1]	0	0

1. Reset value is undefined.

Fields

Field	Function
31 OVR	Overrun Indicates that FLAG generation occurred when FLAG was already 1. Return OVR to 0 by writing 1 to either OVR or FLAG. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overrun 1b - Overrun When writing 0b - No overrun 1b - Return OVR to 0
30-16 —	Reserved
15 OVFL	Overflow Indicates that an overflow has occurred in the internal counter. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>eMIOS_0</td><td>S0–S8
S16
S22–S23</td><td>S9–S15
S17–S21</td></tr>
<tr><td>eMIOS_1</td><td>S0
S8
S16
S22–S23</td><td>S1–S7
S9–S15
S17–S21</td></tr>
<tr><td>eMIOS_2</td><td>S0
S8
S16
S22–S23</td><td>S1–S7
S9–S15
S17–S21</td></tr>
</table>
NOTE
This field behaves differently for register reads and writes.
When reading
0b - No overflow
1b - Overflow
When writing
0b - No overflow
1b - Return OVFL to 0</td></tr>
<tr><td>14-3
—</td><td>Reserved</td></tr>
<tr><td>2
UCIN</td><td>UC Input Pin
Indicates the input pin state after the input pin has been filtered and synchronized.
0b - Negated
1b - Asserted</td></tr>
<tr><td>1
UCOUT</td><td>UC Output Pin
Indicates the output pin state.
0b - Negated
1b - Asserted</td></tr>
<tr><td>0
FLAG</td><td>Flag
Indicates that an input capture or a match event in the comparators has occurred.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - No event 1b - Event has occurred When writing 0b - No event 1b - Clear the flag (FLAG returns to 0)

60.9.6.11 Alternate Address n (ALTA0 - ALTA23)

Offset

For n = 0 to 23:

Register	Offset
ALTAn	34h + (n × 20h)

Function

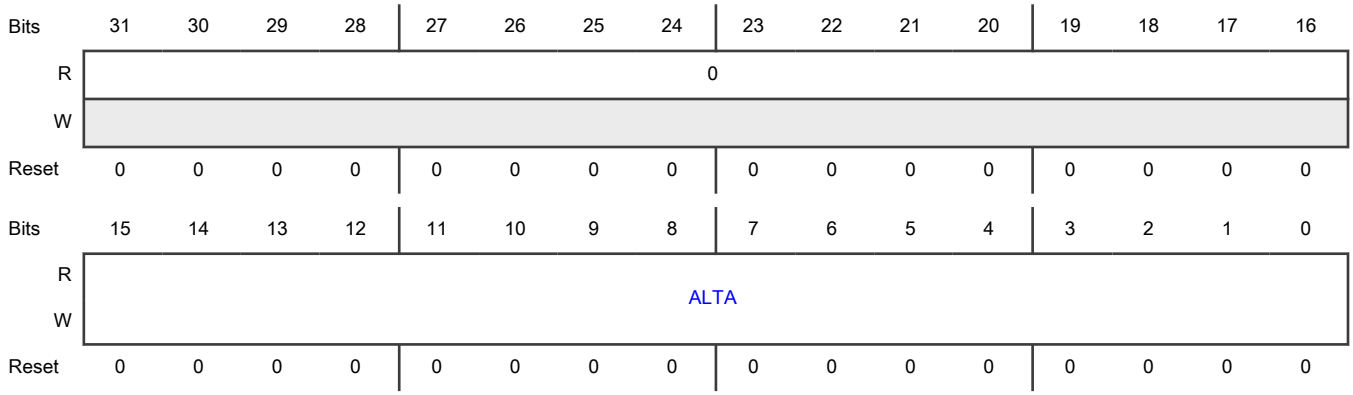
Provides an alternate address to access AS2 shadow registers in these restricted modes:

- General-Purpose Input and Output (GPIO) mode
- Pulse Edge Counting (PEC) mode
- Output PWM with Trigger (OPWMT) mode

In these modes, you can use A*n*[A] together with with ALTA*n*[ALTA] to access both AS1 and AS2. Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA] summarizes the ALTA*n* writing and reading accesses for all ALTA*n* operation modes.

In modes that do not require this register, any write access is ignored and any read access returns unspecified data.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 ALTA	Alternate Address See the register description.

60.9.6.12 UC Control 2 n (C2_0 - C2_23)

Offset

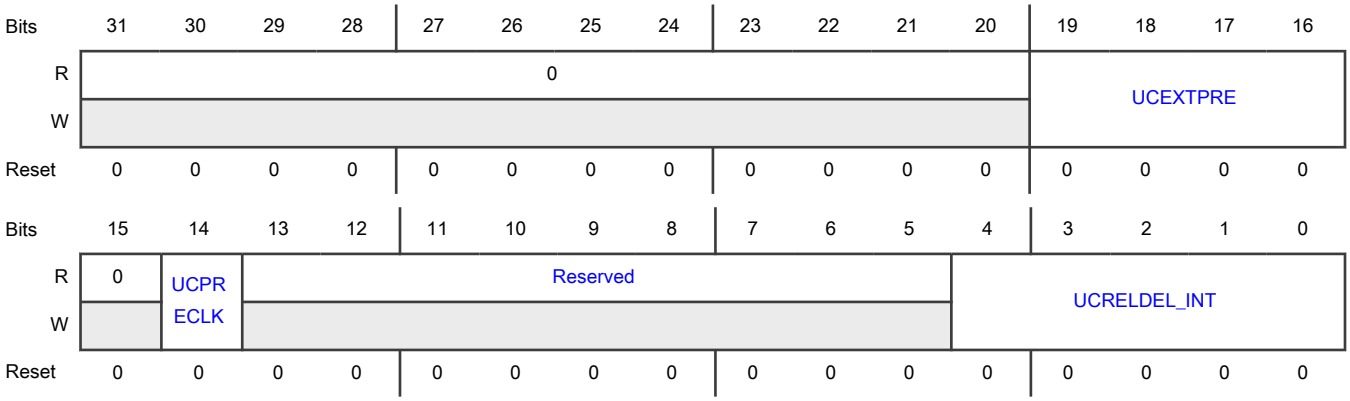
For n = 0 to 23:

Register	Offset
C2_n	38h + (n × 20h)

Function

Provides additional controls for the UC.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 UCEXTPRE	Extended Prescaler Specifies the clock divider for the internal UC prescaler. Write the desired divider value minus 1. For example, for a divider of 1, write 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The two least-significant bits of UCEXTPRE mirror C*n*[UCPRE]. Any write operation to UCEXTPRE also affects UCPRE.
15 —	Reserved
14 UCPRECLK	Prescaler Clock Source Selects the clock source for the internal prescaler. 0b - Prescaled clock 1b - eMIOS module clock
13-5 —	Reserved
4-0 UCRELDEL_INT	Reload Signal Output Delay Interval Specifies the delay interval, in counter bus reload events, between each assertion of AS1-BS1 reload in MC and MCB modes. 00000b - Every event 00001b - Every 2nd event 00010b - Every 3rd event . . . 11111b - Every 32nd event

60.10 Glossary

BIU	IP bus interface unit
CP	Clock prescaler
GCP	Global clock prescaler
IIB	Internal interface bus
PIF	Programmable input filter
UC	Unified channel

Chapter 61
Body Cross-triggering Unit (BCTU)

61.1 Chip-specific BCTU information

61.1.1 BCTU configuration

The following table lists the BCTU configuration.

Table 349. BCTU configuration

Feature	Configuration
FIFO1	16 words
FIFO2	8 words

61.1.2 Instances and trigger sources

This chip has one instance of BCTU module.

The following table lists the BCTU trigger sources.

Table 350. BCTU Trigger sources

BCTU Trigger Number	Module	Source	Applicability
0	eMIOS_0	Channel_0	All
1	eMIOS_0	Channel_1	All
2	eMIOS_0	Channel_2	All
3	eMIOS_0	Channel_3	All
4	eMIOS_0	Channel_4	All
5	eMIOS_0	Channel_5	All
6	eMIOS_0	Channel_6	All
7	eMIOS_0	Channel_7	All
8	eMIOS_0	Channel_8	All
9	eMIOS_0	Channel_9	All
10	eMIOS_0	Channel_10	All
11	eMIOS_0	Channel_11	All
12	eMIOS_0	Channel_12	All
13	eMIOS_0	Channel_13	All
14	eMIOS_0	Channel_14	All
15	eMIOS_0	Channel_15	All
16	eMIOS_0	Channel_16	All
17	eMIOS_0	Channel_17	All

Table continues on the next page...

Table 350. BCTU Trigger sources (continued)

BCTU Trigger Number	Module	Source	Applicability
18	eMIOS_0	Channel_18	All
19	eMIOS_0	Channel_19	All
20	eMIOS_0	Channel_20	All
21	eMIOS_0	Channel_21	All
22	eMIOS_0	Channel_22	All
23	Any of the TRGMUX inp	BCTU_Trg23	All
24	eMIOS_1	Channel_0	All
25	eMIOS_1	Channel_1	All
26	eMIOS_1	Channel_2	All
27	eMIOS_1	Channel_3	All
28	eMIOS_1	Channel_4	All
29	eMIOS_1	Channel_5	All
30	eMIOS_1	Channel_6	All
31	eMIOS_1	Channel_7	All
32	eMIOS_1	Channel_8	All
33	eMIOS_1	Channel_9	All
34	eMIOS_1	Channel_10	All
35	eMIOS_1	Channel_11	All
36	eMIOS_1	Channel_12	All
37	eMIOS_1	Channel_13	All
38	eMIOS_1	Channel_14	All
39	eMIOS_1	Channel_15	All
40	eMIOS_1	Channel_16	All
41	eMIOS_1	Channel_17	All
42	eMIOS_1	Channel_18	All
43	eMIOS_1	Channel_19	All
44	eMIOS_1	Channel_20	All
45	eMIOS_1	Channel_21	All
46	eMIOS_1	Channel_22	All
47	Any of the TRGMUX inp	BCTU_Trg47	All
48	eMIOS_2	Channel_0	MWCT2D17S
49	eMIOS_2	Channel_1	MWCT2D17S

Table continues on the next page...

Table 350. BCTU Trigger sources (continued)

BCTU Trigger Number	Module	Source	Applicability
50	eMIOS_2	Channel_2	MWCT2D17S
51	eMIOS_2	Channel_3	MWCT2D17S
52	eMIOS_2	Channel_4	MWCT2D17S
53	eMIOS_2	Channel_5	MWCT2D17S
54	eMIOS_2	Channel_6	MWCT2D17S
55	eMIOS_2	Channel_7	MWCT2D17S
56	eMIOS_2	Channel_8	MWCT2D17S
57	eMIOS_2	Channel_9	MWCT2D17S
58	eMIOS_2	Channel_10	MWCT2D17S
59	eMIOS_2	Channel_11	MWCT2D17S
60	eMIOS_2	Channel_12	MWCT2D17S
61	eMIOS_2	Channel_13	MWCT2D17S
62	eMIOS_2	Channel_14	MWCT2D17S
63	eMIOS_2	Channel_15	MWCT2D17S
64	eMIOS_2	Channel_16	MWCT2D17S
65	eMIOS_2	Channel_17	MWCT2D17S
66	eMIOS_2	Channel_18	MWCT2D17S
67	eMIOS_2	Channel_19	MWCT2D17S
68	eMIOS_2	Channel_20	MWCT2D17S
69	eMIOS_2	Channel_21	MWCT2D17S
70	eMIOS_2	Channel_22	MWCT2D17S
71	Any of the TRGMUX inp	BCTU_Trg71	MWCT2D17S

NOTE

You must clear triggers to BCTU after TRGMUX programming using TRGCFG_n[TRG_FLAG] bit.

61.2 Introduction

BCTU accepts ADC conversion-request trigger inputs and routes those requests to one or more ADCs.

BCTU accepts the following trigger sources:

- Hardware triggers (outputs from on-chip timer modules)
- Software triggers through the SFTRGRn register

BCTU maps each trigger input to a Trigger Configuration (TRGCFG_0 - TRGCFG_71) register, which specifies the target ADC channel and other control data for the requested conversion.

You can configure a list of ADC channels, the CL, that processes a sequence of conversions in response to a single trigger input. See the Conversion list (CL) section.

BCTU stores each ADC conversion result for CPU access or routing to memory through DMA.

If the same ADC is targeted by multiple triggers, priority logic selects the next trigger to process from all active trigger inputs. See Priority of simultaneous triggers.

BCTU inputs are active-high signals. That is, a logic 1 asserts the trigger input.

61.3 Features

BCTU supports:

- 72 trigger inputs, configured with the Trigger Configuration (TRGCFG_0 - TRGCFG_71) registers
- Trigger outputs to 3 ADCs
- One priority selection logic block per ADC
- Selection multiplexers to select one trigger at a time for each ADC
- A CL of 32 ADC channels
- A trigger grouping block to latch the hardware triggers. BCTU clears these triggers when the ADC sends an acknowledgement for conversion completion.

61.4 Block diagram

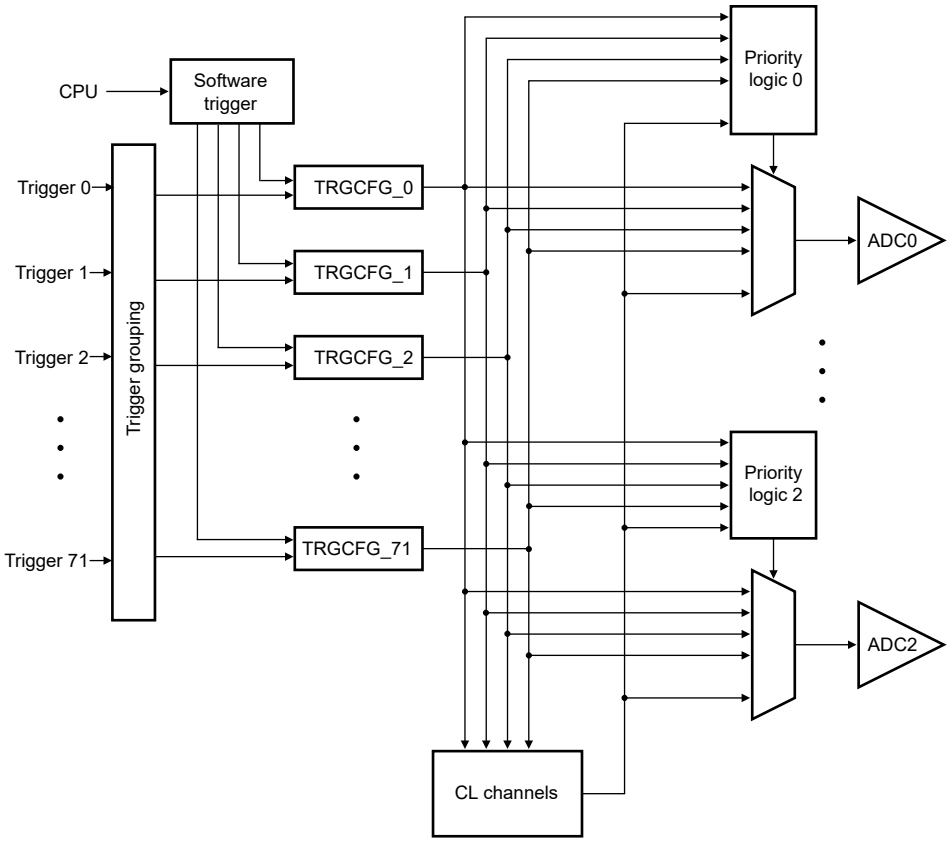


Figure 345. BCTU block diagram

61.5 Interaction with other peripherals

BCTU works together with timer and ADC modules on the chip. Use the Trigger Configuration registers (TRGCFG_*n*) to map BCTU triggers to ADC inputs. When a timer output asserts a BCTU trigger, BCTU passes the trigger to an ADC input and stores

the trigger assertion until the corresponding ADC begins the requested conversion. When the ADC completes the conversion, BCTU asserts a trigger clear signal to clear both the internally stored trigger and the external trigger source if they are still asserted. Implementation illustrates the modules that interact with BCTU.

NOTE

A timer trigger can remain active until the corresponding ADC initiates conversion. Therefore, another match may occur in the timer before the ADC conversion begins. BCTU does not detect this trigger overrun. Your application must look for and respond to such conditions.

61.6 Implementation

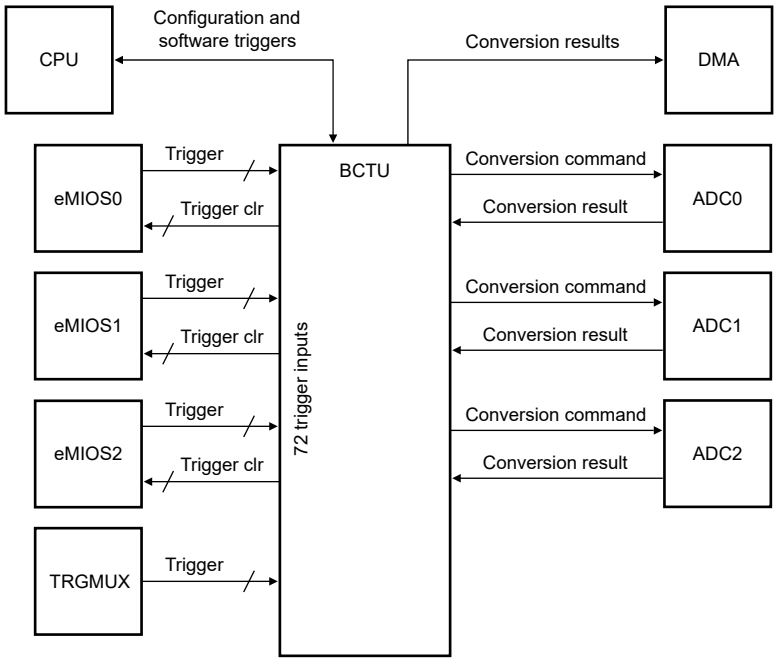


Figure 346. Implementation

61.7 Module Disable (Low Power state)

Module Disable (MCR[MDIS]) stops the BCTU clock, resulting in reduced power consumption.

When you disable BCTU (write 1 to MDIS), it:

- Waits for the completion of any in-progress conversions before stopping the clock.
- Ignores all trigger inputs.
- Does not trigger any ADC conversion requests.
- Maintains any asserted trigger inputs, which you can clear if necessary.

When you re-enable BCTU (write 0 to MDIS) and Global Triggers are enabled (MCR[GTRGEN]), BCTU processes all active asserted trigger inputs and sends conversion requests to ADCs according to the priority scheme.

61.8 Software reset

You can reinitialize all BCTU registers to their reset values with Software Reset (MCR[Software_Reset]).

Software reset does not clear any active trigger inputs, but it does put the ADC interface into the Reset state. Therefore, to avoid data incoherency, you must terminate any BCTU-triggered conversions or wait until they are complete before asserting software reset.

61.9 Triggers

61.9.1 Overview

BCTU prioritizes ADC trigger inputs based on the input number: the lower the number, the higher the priority associated with it. Therefore, trigger 0 is the highest priority trigger and trigger 71 is the lowest priority. The trigger input priority scheme determines which trigger takes precedence when more than one trigger occurs at the same time, or when the triggered ADC is busy with a conversion. Therefore, there are two possible scenarios for the trigger priority selection:

- When there are no ongoing conversions
- When a triggered ADC is executing a conversion

BCTU supports two types of trigger sources:

- Hardware triggers from timer modules
- Software triggers (SFTRGR*n*)

61.9.2 Hardware triggers

To request an ADC conversion through the BCTU, a timer module such as eMIOS asserts its output signal, which is connected to a BCTU trigger input. When BCTU detects the asserted trigger, it routes that request to the ADCs selected in the corresponding Trigger Configuration (TRGCFG_*n*[ADC_SEL*m*]). The trigger remains asserted until the requested conversion begins, at which time BCTU clears the trigger. If the timer asserts the trigger input when the trigger is already set, BCTU ignores the new trigger. If BCTU receives a new trigger input at the same time it is clearing the trigger, then the trigger remains asserted and BCTU initiates a new conversion.

If you configure the trigger for CL, trigger assertion begins a sequence of conversions controlled by the CL Channel Address (LISTCHR_0 - LISTCHR_15) registers. In this event, BCTU clears the trigger when the last conversion in the CL begins.

NOTE

Any new trigger input participates in the trigger priority selection scheme and therefore may not trigger the next conversion.

61.9.3 Software triggers

You assert a software trigger by writing 1 to the corresponding field of the appropriate Software Trigger (SFTRGR*n*) register. Similar to the hardware triggers, BCTU changes the software trigger field to 0 when the requested conversion begins. If you write 1 to the software trigger field while the conversion is still in progress, BCTU ignores the request. If you write 1 to the field at the same time BCTU is attempting to clear the trigger (change the field to 0), the trigger remains asserted and BCTU triggers a new conversion following the trigger priority criteria.

61.9.4 Configure all triggers

Global Trigger Enable (MCR[GTRGEN]) allows you to configure the BCTU trigger inputs and then activate them all at one time.

To configure and then activate all BCTU triggers:

1. Clear any currently active and asserted trigger inputs (write 0 to TRGFG_*n*[TRG_FLAG]), which also causes BCTU to assert trigger clear to the trigger sources.
2. Disable Global Triggers (MCR[GTRGEN]). This action deactivates all trigger inputs and prevents BCTU from generating any ADC conversion requests.
3. Configure the individual trigger inputs (see Configure individual triggers).

4. Enable Global Triggers (MCR[GTRGEN]). This action activates all individually enabled triggers at the same time.

61.9.5 Configure individual triggers

Using Trigger Configuration (TRGCFG_0 - TRGCFG_71), for each trigger:

- Enable the trigger (TRIGEN).
- Specify the target ADC (ADC_SEL*n*).
- Specify the ADC channel (CHANNEL_VALUE_OR_LADDR).
- Specify whether the trigger initiates a single conversion or a CL (TRS).
- For a CL trigger, specify a pointer to the CL element (TRGCFG_x[CHANNEL_VALUE_OR_LADDR]).

61.9.6 Reconfigure triggers on-the-fly

NOTE

Reconfigure BCTU triggers only when the individual trigger is not active (TRG_FLAG). If TRG_FLAG is 1, the requested conversion is still in progress and it is not safe to modify the trigger configuration.

To reconfigure a trigger on-the-fly (while BCTU is in full operation):

1. Disable the trigger (TRIGEN) to prevent it from requesting a new conversion.
2. Check the current state of the trigger (TRG_FLAG) to be sure no conversion or CL is in progress.
3. When any in-progress conversion or CL completes (TRG_FLAG == 0), clear the trigger source by writing 1 to TRG_FLAG. This action ensures that only trigger inputs that assert after you reconfigure and re-enable the trigger are able to request a new conversion or CL.
4. Reconfigure the trigger (see Configure individual triggers).
5. Enable the trigger (TRIGEN).

61.9.7 Enable software triggers

To allow software triggers (SFTRGR*n*[SFTRG*m*]) to request conversions, you must enable Global Trigger Enable (MCR[GTRGEN]). If you disable Global Trigger Enable, software triggers do not initiate a conversion but asserted triggers remain asserted. However, when you enable Global Trigger Enable, any asserted software triggers initiate conversions following the priority scheme of lowest to highest trigger number.

Software triggers operate regardless of individual Trigger Enable (TRGCFG_x[TRIGEN]). That is, writing 1 to any Software Trigger (SFTRGR*n*[SFTRG*m*]) initiates a conversion (following the priority scheme) regardless of the value of Trigger Enable. This software trigger independence supports the elimination of conflict between software and hardware triggers.

61.9.8 Trigger timing

There are four clock cycles from assertion of a BCTU trigger input to assertion of the trigger output to the ADC. For single conversions, BCTU asserts Trigger Clear back to the trigger source simultaneous to asserting the trigger to the ADC.

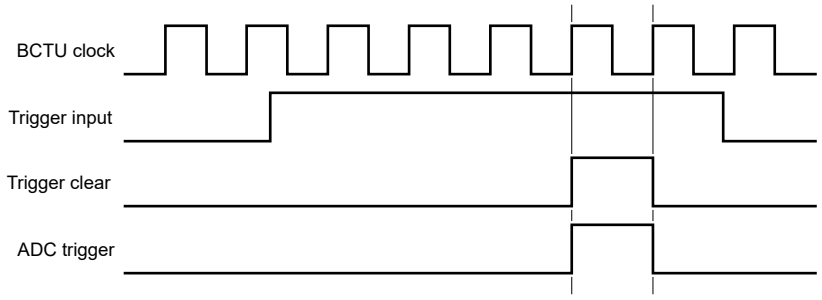


Figure 347. Trigger timing

61.9.9 Priority of simultaneous triggers

When more than one trigger input asserts a conversion request to the same ADC at the same time—that is, in the same BCTU clock cycle—BCTU employs a priority scheme to determine which asserted trigger to process next. BCTU prioritizes trigger inputs based on the trigger number. The lowest number, trigger 0, has the highest priority, and the highest number, trigger 71, has the lowest priority.

If two triggers assert in the same clock cycle, BCTU gives priority to the lower numbered trigger. For example, if trigger 0 and trigger 1 both assert in the same cycle, BCTU selects trigger 0 for trigger generation. Trigger 1 remains active (pending) until the conversion requested by trigger 0 completes. See Simultaneous trigger priority management.

If trigger 0 asserts again before the next trigger selection opportunity at the end of the current conversion, then BCTU selects it again and trigger 1 remains asserted (pending) during the next ADC conversion time frame.

As illustrated in Trigger timing, BCTU requires four clock cycles to transmit a conversion request to an ADC if that ADC is not already executing a conversion. However, if the ADC is busy at the time of trigger assertion, BCTU requires only three clock cycles to generate the conversion request for a subsequent conversion.

Triggers that assert in the same clock cycle but target different ADCs are not in conflict, and BCTU processes both triggers at the same time.

61.9.10 Simultaneous trigger priority management

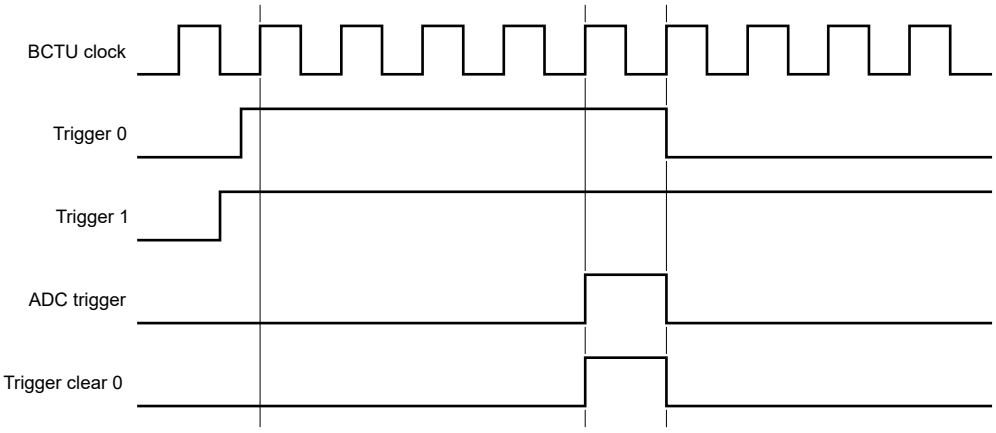


Figure 348. Simultaneous trigger priority management

61.9.11 Trigger management when ADC is busy

When a trigger input asserts a conversion request to an ADC that is already processing a conversion requested by a different trigger, the new trigger remains active (pending) until the conversion in progress completes. This behavior is the same as if the two triggers had been simultaneous (occurred in the same clock cycle). When the in-progress conversion completes, the pending trigger then competes for priority with any other active trigger for that ADC.

61.10 Conversion list (CL)

61.10.1 CL overview

A CL executes a sequence of conversions on one or more target ADCs. See CL architecture and operation. It also stores information from the last conversion executed in a sequence except for the target ADC, which is stored in ADC Select n (TRGCFG_*m*[ADC_SEL*n*]).

If the Channel or CL Address (TRGCFG_x[CHANNEL_VALUE_OR_LADDR]) points to a CL element containing one or more entries:

1. The trigger initiates a sequence of conversions starting with the target CL element.
2. The sequence executes until the LAST channel (LISTCHR_*n*[LAST_y] is 1), and stops after that channel is converted.

When you configure the CL element for Next Channel Wait For Trigger (write 1 to LISTCHR_*n*[NEXT_CH_WAIT_ON_TRIG_y]):

1. BCTU clears the trigger source.
2. CL execution pauses after the current conversion is completed.
3. CL execution resumes when the same trigger asserts again.

BCTU determines trigger priority before executing any CL. See Priority of simultaneous triggers.

61.10.2 CL architecture and operation

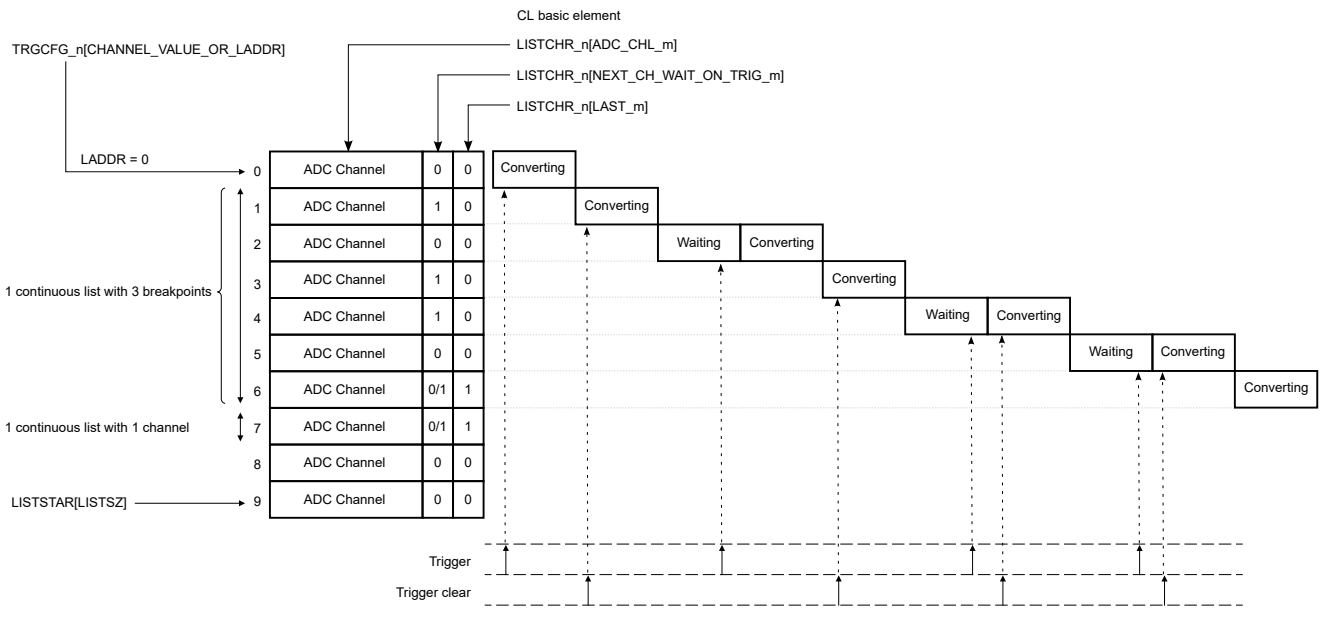


Figure 349. CL architecture and operation

61.10.3 CL channel rules

CL conversion obeys the following rules:

- The sequence of conversions executes until the Last Channel (LISTCHR_*n*[LAST_y] is 1).
- A CL conversion sequence precludes other conversions on the same ADC. If the CL is waiting (LISTCHR_*n*[NEXT_CH_WAIT_ON_TRIG_y] is 1), BCTU cannot route a non-CL conversion to that ADC (although that ADC remains available for normal and injected conversions).
- Two or more triggers for different ADCs may point to the same CL. This CL can drive parallel execution on the selected ADCs.
- If the Channel or CL Address (TRGCFG_x[CHANNEL_VALUE_OR_LADDR]) does not point to a valid list address, BCTU uses address 0 by default.
- When the CL pointer reaches the last available list address (LISTSTAR[LISTSZ] minus 1), the pointer wraps to list address 0.
- You can select more than one ADC per trigger using the ADC Select n (TRGCFG_x[ADC_SELn]) fields, but selecting more than one ADC is valid only for multiple parallel CL operation. If you select more than one ADC for a single-conversion trigger, BCTU ignores the trigger.

61.11 ADC conversion results access

61.11.1 Overview

The value of DATA_DEST determines access to ADC results.

- DATA_DEST = 000: ADC results are available in the ADC data registers.
- DATA_DEST = 001: ADC results are available in FIFO 1 Result Data (FIFO Result Data (FIFO1DR - FIFO2DR)). See Route conversion data to FIFO.
- DATA_DEST = 010: ADC results are available in FIFO 2 Result Data (FIFO Result Data (FIFO1DR - FIFO2DR)). See Route conversion data to FIFO.

You can access ADC results using flags and interrupt requests (see Access on interrupt request) or DMA (see Access on DMA request). Interrupt requests are based on Module Status (MSR) flags. ADC*n*DR provides the following information, with one register used for each ADC:

- The conversion results
- The channel that was used for the conversion
- The trigger input that initiated the conversion
- Whether a CL element generated the conversion
- Whether the data was the last conversion of a CL

61.11.2 Access on interrupt request

If interrupts are enabled for ADC*n* Result Data (MCR[IEN*n*] is 1) and DMAs are disabled for ADC*n* Result Data (MCR[DMA*n*] is 0), BCTU generates an interrupt request for each available conversion data in ADC*n*DR. Because BCTU executes conversions on different ADCs in parallel, conversion results may be available on multiple ADCs at the same time. New Data (MSR[NDATAn] is 1) indicates that new conversion data is available in the corresponding ADC Data Result (ADC*n*DR). To clear the New Data flag, write 1 to MSR[NDATAn_CLR].

61.11.3 Access on DMA request

If you enable DMA requests (MCR[DMA*n*]):

- You must select one channel for DMA per ADC.
- New conversion result data (ADC*n*DR) generates a DMA request.

- The DMA controller deasserts the DMA request and writes 1 to MSR[NDATAn_CLR] to clear the flag.

When an ADC generates New Data (MSR[NDATAn]), BCTU processes the DMA request. See Clear a DMA request.

61.11.4 Clear an interrupt request

BCTU clears an interrupt request when:

- The number of valid entries is less than or equal to the FIFOn Watermark Level (FIFOWM[WM_FIFO*n*]).
- You write 1 to FIFO Watermark Interrupt Status (FIFOERR[WM_INT_FIFO*n*]).

61.11.5 Clear a DMA request

BCTU clears a DMA request when:

- The number of valid entries is less than or equal to the FIFOn Watermark Level (FIFOWM[WM_FIFO*n*]).
- The DMA controller sends acknowledgment to BCTU.

61.11.6 Clear an error-generated interrupt request

Clear an interrupt generated because of an error condition by writing 1 to the appropriate error flag:

- FIFOERR[OVR_ERR_FIFO1]
- FIFOERR[UNDR_ERR_FIFO1]
- FIFOERR[OVR_ERR_FIFO2]
- FIFOERR[UNDR_ERR_FIFO2]

61.11.7 Conversion data overrun

If new ADC conversion data becomes available before you read the previous data (ADC*n*DR), then BCTU indicates a data overrun (MSR[DATAOVRn] is 1).

If you enable interrupts for the ADC (MCR[IEN*n*]), a data overrun also generates an interrupt request, regardless of the DMA request configuration.

To clear a data overrun (MSR[DATAOVRn]), write 1 to Data Overrun Clear (MSR[DATAOVRn_CLR]). See Clear an error-generated interrupt request.

61.11.8 Route conversion data to FIFO

BCTU allows you to route multiple ADC conversion results to one of 2 internal data FIFOs. Any trigger source can route results to an available FIFO based on the trigger index. Each FIFO has a dedicated counter (FIFOCNTR[CNTR_FIFO*n*]) to indicate the number of entries in the FIFO.

Each FIFO also has its own result data register (FIFO*n*DR) that contains the following information about the last conversion result routed to the FIFO.

- ADC result data
- Trigger source index
- ADC number
- ADC channel number

61.11.9 Read ADC result from FIFO

Each FIFO has DMA and interrupt interfaces. The DMA interface reads data. The interrupt interface indicates watermark and error conditions:

- An overrun error (FIFOERR[OVR_ERR_FIFO*n*]) occurs when you attempt to write data to an already full FIFO.
- An underrun error (FIFOERR[UNDR_ERR_FIFO*n*]) occurs when you attempt to read data from an empty FIFO.
- BCTU asserts a Watermark interrupt request if the number of valid FIFO entries exceeds the watermark level (FIFOWM[WM_FIFO*n*]).

You can read a FIFO:

- by enabling DMA requests (FIFOCR[DMA_EN_FIFO*n*])
- by polling the FIFO Full status (FIFOSR[FULL_FIFO*n*])

See Configuration of DMA and interrupt behavior.

61.11.10 Configuration of DMA and interrupt behavior

Table 351. Configuration of DMA and interrupt behavior

FIFOCR[DMA_EN_FIFO*n*]	FIFOCR[IEN_FIFO*n*]	Condition
0	0	No DMA request; no interrupt request
0	1	Interrupt request when FIFO exceeds watermark level; interrupt request on error
1	0	DMA request when FIFO exceeds watermark level
1	1	DMA request when FIFO exceeds watermark level; interrupt request on error

61.12 Multiple parallel conversions (MPC)

61.12.1 Overview

The MPC mechanism allows you to execute a series of simultaneous ADC conversions by selecting more than one ADC for any CL-configured trigger.

61.12.2 Configure MPC

To configure a CL to perform MPC:

1. Select CL for the trigger resolution (write 1 to TRGCFG_*m*[TRS]).
2. Select more than one ADC for the trigger (TRGCFG_*m*[ADC_SEL*n*]).
3. Specify the CL address for the first CL element (TRGCFG_*m*[CHANNEL_VALUE_OR_LADDR]).

61.12.3 CL address selection for MPC

When the trigger input for the CL asserts, BCTU uses the Channel or List Address (CHANNEL_VALUE_OR_LADDR) as an index into the CL for the lowest numbered ADC selected (ADC_SEL*n*), and each subsequent element of the CL for each remaining selected ADC. The set of selected ADC channels perform their conversions in parallel. When that set of conversions is complete, the next set of CL elements determine the next set of ADC channels to perform parallel conversions, as illustrated in:

- Example: MPC with two ADCs
- Example: MPC with three ADCs

61.12.4 Example: MPC with two ADCs

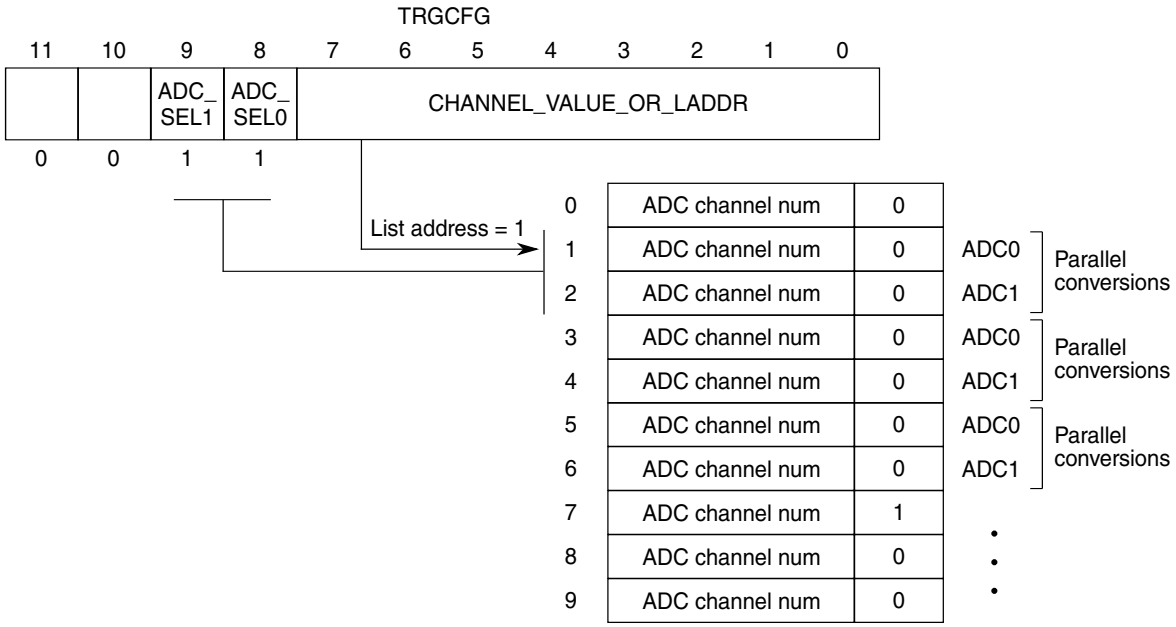


Figure 350. Example: MPC with two ADCs

61.12.5 Example: MPC with three ADCs

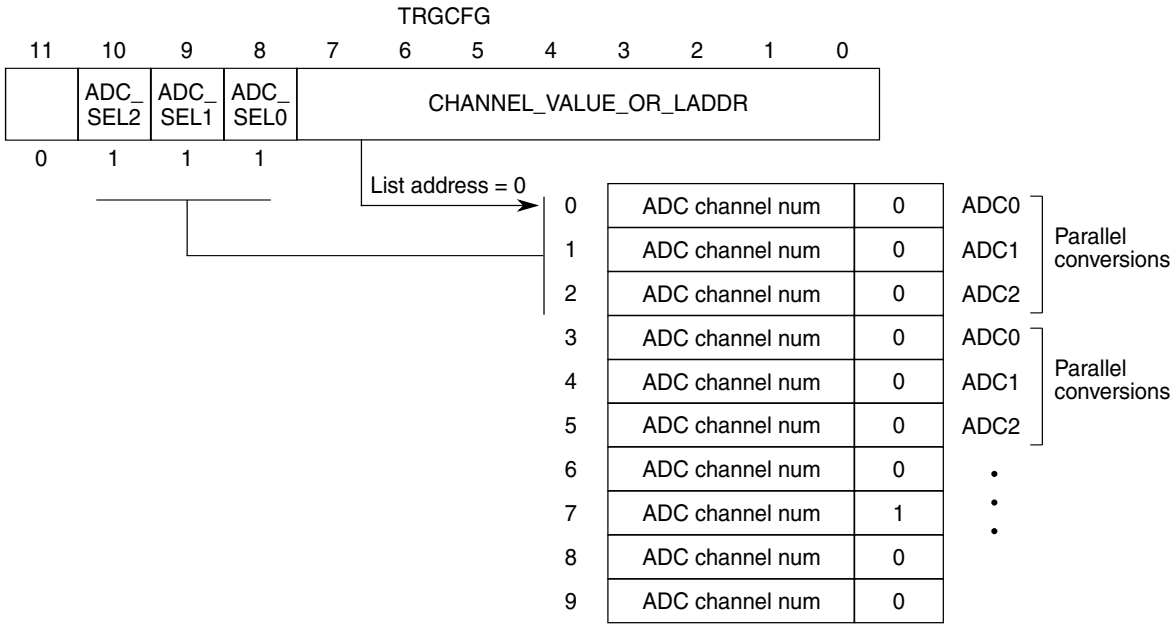


Figure 351. Example: MPC with three ADCs

61.12.6 MPC priority rules

MPC follows the same priority rules as for single conversions, which is based on the trigger input number: the lower the trigger, the higher the priority. However, MPC cannot start until all ADCs selected for the CL are available; that is, they are not currenlty

engaged in conversion. For example, if a trigger input asserts for a CL using ADC0 and ADC1, BCTU does not initiate the CL until ADC0 and ADC1 are both available.

Also, if ADC0 is available but there is a higher priority trigger for ADC1, the MPC does not start. If trigger inputs assert for two MPCs that have at least one ADC in common, the lower trigger number has higher priority. A priority conflict occurs only when separate requested CLs have at least one ADC in common. If the MPCs use different ADCs, there is no conflict and the MPCs execute at the same time.

61.13 Interrupts and DMA requests

Table 352. Interrupts and DMA requests

Event	Enable interrupt	Enable DMA request	Description
Trigger Flag	MCR[TRGEN]	—	Interrupt request on assertion of Trigger Flag (MSR[TRGF])
ADC Data Ready	MCR[IEN*n*]	MCR[DMA*n*]	Interrupt or DMA request when ADC result data is available (ADC*n*DR)
CL Last	MCR[LIST_IEN]	—	Interrupt request when the last conversion of any CL executes
FIFO Watermark	FIFOCR[IEN_FIFO*n*]	FIFOCR[DMA_EN_FIFO*n*]	Interrupt or DMA request when the number of valid FIFO entries exceeds the FIFO

61.14 BCTU register descriptions

All BCTU registers are 32 bits wide. 8-bit write operations are allowed. The BCTU module does not include any coherence mechanism, so application software must ensure coherency.

NOTE

- You must always write 0 to reserved writable registers and fields. Read accesses of reserved registers and fields do not return meaningful data.
- For registers with write protection, if protection is active and you attempt to write to those registers, register content is unaffected and BCTU issues a transfer error.

61.14.1 BCTU memory map

BCTU base address: 4008_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration (MCR)	32	RW	0000_0000h
8h	Module Status (MSR)	32	RW	0000_0000h
18h - 134h	Trigger Configuration (TRGCFG_0 - TRGCFG_71)	32	RW	0000_0000h
228h	Write Protection (WRPROT)	32	RW	0000_0000h
22Ch	Software Trigger 1 (SFTRGR1)	32	RW	0000_0000h
230h	Software Trigger 2 (SFTRGR2)	32	RW	0000_0000h
234h	Software Trigger 3 (SFTRGR3)	32	RW	0000_0000h
23Ch - 244h	ADCn Result Data (ADC0DR - ADC2DR)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
24Ch	CL Size Status (LISTSTAR)	32	R	0000_0020h
250h - 28Ch	CL Channel Address (LISTCHR_0 - LISTCHR_15)	32	RW	0000_0000h
450h - 454h	FIFO Result Data (FIFO1DR - FIFO2DR)	32	R	0000_0000h
460h	FIFO Control (FIFOCR)	32	RW	0000_0000h
464h	FIFO Watermark Configuration (FIFOWM)	32	RW	0000_0000h
468h	FIFO Error/Status (FIFOERR)	32	RW	0000_0000h
46Ch	FIFO Status (FIFOSR)	32	R	0000_0000h
470h	FIFO Counter (FIFOCNTR)	32	R	0000_0000h

61.14.2 Module Configuration (MCR)

Offset

Register	Offset
MCR	0h

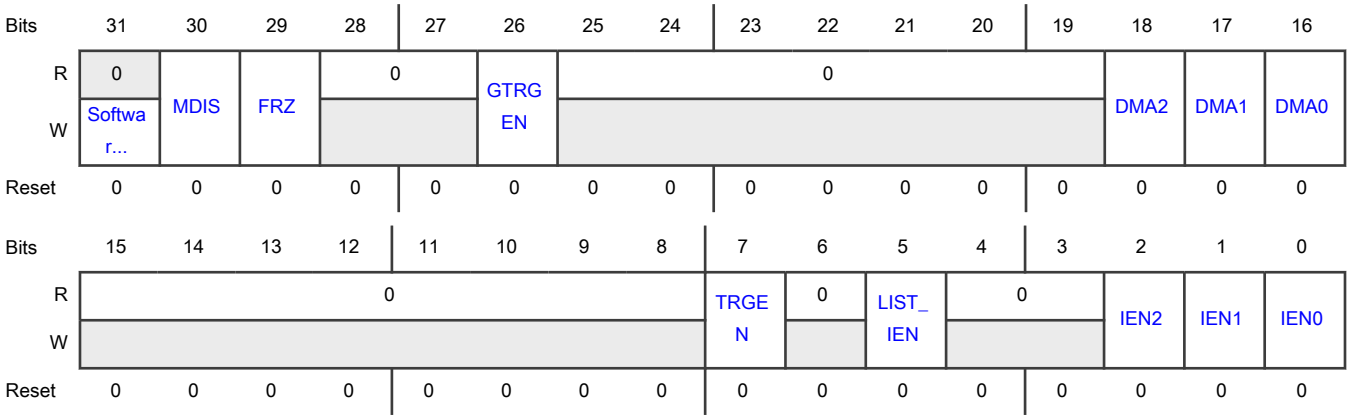
Function

Configures the following aspects of BCTU behavior:

- Low power
- Debug
- Trigger control
- DMA control
- Interrupt control

Access: User read/write

Diagram



Fields

Field	Function
31 Software_Reset	Software Reset Asserts soft reset, which initializes all BCTU registers and internal state machines to their reset values. 0b - Deasserts 1b - Asserts
30 MDIS	Module Disable Disables BCTU by putting it in Low Power state (see Module Disable (Low Power state)). 0b - Enable (normal operation) 1b - Disable (low-power operation)
29 FRZ	Debug Freeze Disables all hardware trigger inputs but leaves the software trigger enabled. Debug Freeze isolates BCTU from external triggers and allows you to manually trigger a conversion and read the conversion result. 0b - Disables 1b - Enables
28-27 —	Reserved
26 GTRGEN	Global Trigger Enable Enables all individually enabled trigger inputs at the same time. For any given input trigger to be active, you must both enable the individual trigger (TRGCFG_*n*[TRIGEN]) and Global Trigger Enable. See Configure all triggers. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-19 —	Reserved
18-16 DMAn	Enable ADCnDR DMA Enables DMA operation when new data is available in ADC*n*DR. 0b - Disable 1b - Enable
15-8 —	Reserved
7 TRGEN	Trigger Interrupt Request Enable Enables an interrupt request on a Trigger Flag (MSR[TRGF]). 0b - Disable 1b - Enable
6 —	Reserved
5 LIST_IEN	CL Interrupt Enable Enables an interrupt request for the last conversion in a CL (both ADC*n*DR[LIST] and ADC*n*DR[LAST] are 1). 0b - Disable 1b - Enable
4-3 —	Reserved
2-0 IENn	Interrupt Enable For ADCnDR Enables an interrupt request when BCTU writes a new conversion result to ADC Data (ADC*n*DR). Also enables an interrupt request on an overrun in ADC*n*DR, which indicates the previous data has been overwritten by a new conversion result. See also MSR[NDATA*n*] and MSR[DATAOVR*n*]]. Enabling ADC*n*DR DMA (DMA*n* field) disables the interrupt for new data in ADC*n*DR, but the interrupt for ADC*n*DR data overrun remains enabled. 0b - Disable 1b - Enable

61.14.3 Module Status (MSR)

Offset

Register	Offset
MSR	8h

Function

Indicates the status of various BCTU functions, including:

- Whether conversion data is available for each ADC
- CL operation, such as the last conversion command executed in a CL
- Whether BCTU triggered any of the ADCs

All flag fields become 0 upon assertion of a low-power (software) or clock-stop (hardware) request. You can turn off individual flags by writing 1 to the field.

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0				0	0	0	0	0	0	0	0	0	0	0
W	TRGF_CLR					LIST2_L...	LIST1_L...	LIST0_L...		DATAOVR...	DATAOVR...	DATAOVR...		NDATA2_...	NDATA1_...	NDATA0_...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TRGF	0				LIST2_L...	LIST1_L...	LIST0_L...	0	DATAOVR2	DATAOVR1	DATAOVR0	0	NDATA2	NDATA1	NDATA0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TRGF_CLR	TRGF Clear Changes Trigger Flag (TRGF) to 0. 0b - No action 1b - Changes to 0
30-27 —	Reserved
26-24	CL n Last Clear

Table continues on the next page...

Table continued from the previous page...

Field	Function
LISTn_Last_CLR	Changes CL *n* Last (LIST*n*_Last) to 0. 0b - No action 1b - Changes to 0
23 —	Reserved
22-20 DATAOVRn_CLR	DATAOVRn Clear Changes Data Overrun *n* (DATAOVR*n*) to 0. 0b - No action 1b - Changes to 0
19 —	Reserved
18-16 NDATAn_CLR	New Data Clear Changes New Data *n* (NDATA*n*) to 0. 0b - No action 1b - Changes to 0
15 TRGF	Trigger Flag Indicates at least one ADC was triggered. 0b - No ADC triggered 1b - An ADC was triggered
14-11 —	Reserved
10-8 LISTn_Last	CL n Last Conversion Indicates that ADCn has executed the last conversion in a CL. 0b - Last conversion not complete 1b - Last conversion complete
7 —	Reserved
6-4 DATAOVRn	Data Overrun n Indicates that the data in ADCn has been overwritten with new data. 0b - Data not overwritten 1b - Data overwritten

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 —	Reserved
2-0 NDATAn	New Data n Indicates that new conversion data is available for ADCn. 0b - Not available 1b - Available

61.14.4 Trigger Configuration (TRGCFG_0 - TRGCFG_71)

Offset

For a = 0 to 71:

Register	Offset
TRGCFG_a	18h + (a × 4h)

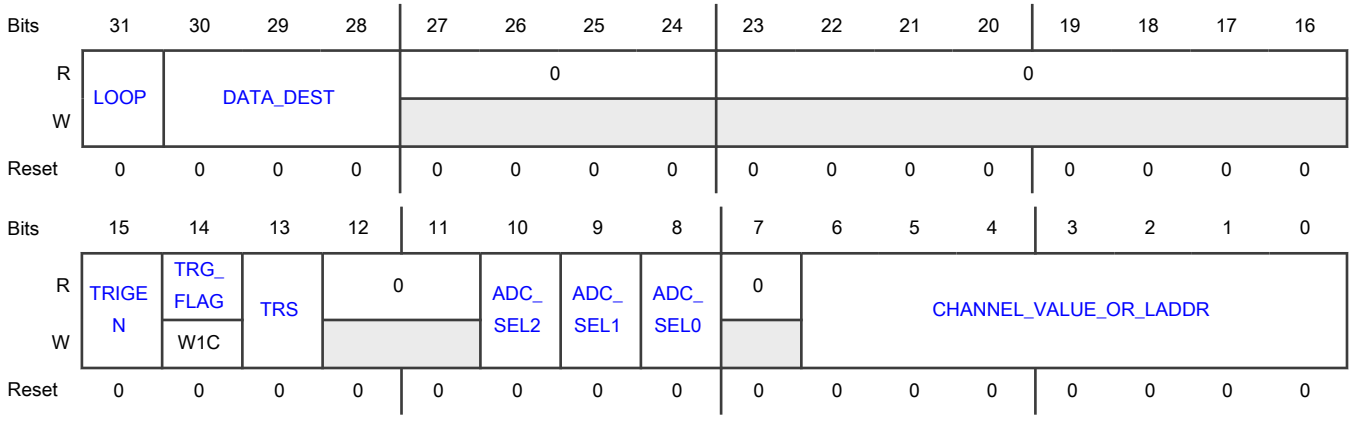
Function

Configures the corresponding trigger:

- Specifies the target ADC and ADC channel to use for conversion.
- Specifies whether to use a CL or a single ADC conversion command.

Access: User read/write

Diagram



Fields

Field	Function
31 LOOP	Loop Specifies that the current trigger executes in a loop; that is, the triggered conversion repeats until you disable Loop. Each loop-triggered conversion wins the priority evaluation among all pending triggers, and executes only if it is the highest-priority triggered conversion at the moment the conversion starts. 0b - Disable 1b - Enable
30-28 DATA_DEST	Data Destination Specifies the destination for storing the conversion results. You can route each ADC's conversion result to either of the following: • ADC-specific data registers • One of the available FIFOs BCTU routes conversion results to different destinations based on the trigger index that requested the conversion. You cannot duplicate the same data in more than one destination. 000b - ADC-specific data registers 001b - FIFO1 010b - FIFO2 011b - Reserved All other values are reserved.
27-24 —	Reserved
23-16 —	Reserved
15 TRIGEN	Trigger Enable Enables hardware input triggers to initiate ADC conversions. This field does not affect software triggers. 0b - Disable 1b - Enable
14 TRG_FLAG	Trigger Flag Indicates the trigger input is asserted and an ADC conversion is in progress. When the conversion completes, BCTU changes this field to 0. Writing 1 to this field causes the input trigger event to clear when the in-progress conversion completes, and prevents this trigger from initiating a new ADC conversion until the input trigger is asserted again. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No ongoing ADC conversion initiated 1b - Ongoing ADC conversion initiated When writing 0b - No action 1b - Changes to 0
13 TRS	Trigger Resolution Selects single conversion or CL conversions. 0b - Single conversion 1b - CL conversions
12-11 —	Reserved
10-8 ADC_SELn	ADC Select n Selects ADC *n* for conversion. 0b - Deselects 1b - Selects
7 —	Reserved
6-0 CHANNEL_VALUE_OR_LADDR	Channel or CL Address Depending on the value of TRS: • 0: Specifies the ADC channel for conversion • 1: Specifies the address of the first CL position to execute

61.14.5 Write Protection (WRPROT)

Offset

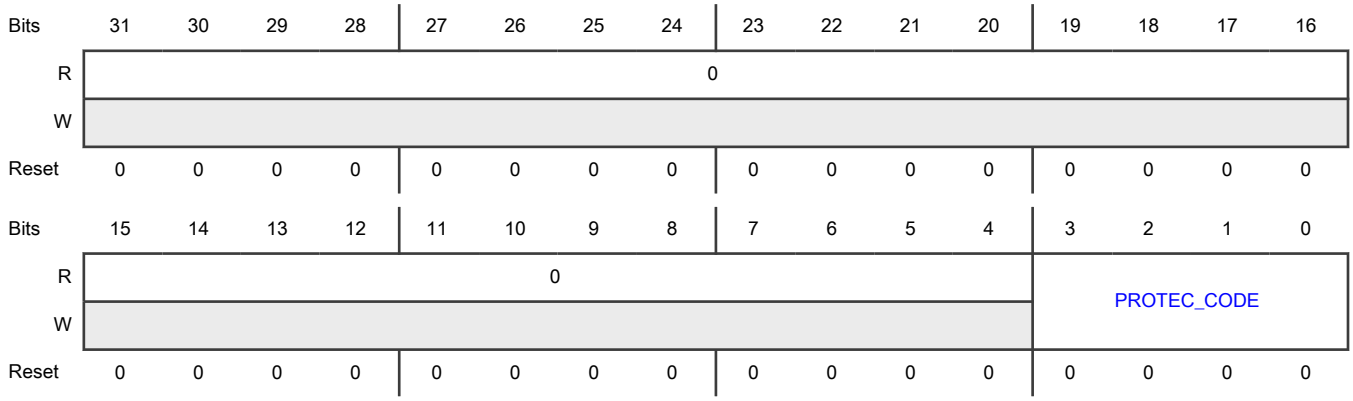
Register	Offset
WRPROT	228h

Function

Enables write protection at reset exit. To allow writes to protected registers, you must write a special code into PROTEC_CODE. There are two codes to remove protection: one that permanently disables the protection; one that disables protection during one write cycle to a protected register, after which the protection code returns to the initial (protected) value of 0000b.

Access: User read/write

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 PROTEC_CODE	Protection Code Controls the protection of write-protected registers. **NOTE** You can write this field only in Normal mode (MCR[MDIS] is 0). 0000b - Enable protection 1001b - Disable protection for one write cycle 1010b - Disable protection permanently

61.14.6 Software Trigger 1 (SFTRGR1)

Offset

Register	Offset
SFTRGR1	22Ch

Function

Allows you to trigger conversions on the associated triggers.

To use a software trigger, you must first disable the corresponding hardware trigger (TRGCFG_x[TRIGEN]).

BCTU writes 0 to a software trigger field on completion of the triggered conversion.

Access: User read/write; this register is write protected by Write Protection (WRPROT). If protection is enabled, writes to this register generate a transfer error and do not modify its content.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SFTRG31	SFTRG30	SFTRG29	SFTRG28	SFTRG27	SFTRG26	SFTRG25	SFTRG24	SFTRG23	SFTRG22	SFTRG21	SFTRG20	SFTRG19	SFTRG18	SFTRG17	SFTRG16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SFTRG15	SFTRG14	SFTRG13	SFTRG12	SFTRG11	SFTRG10	SFTRG9	SFTRG8	SFTRG7	SFTRG6	SFTRG5	SFTRG4	SFTRG3	SFTRG2	SFTRG1	SFTRG0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SFTRGm	Software Trigger m Starts an ADC conversion. 0b - No effect 1b - Trigger conversion

61.14.7 Software Trigger 2 (SFTRGR2)

Offset

Register	Offset
SFTRGR2	230h

Function

Allows you to trigger conversions on the associated triggers.

To use a software trigger, you must first disable the corresponding hardware trigger (TRGCFG_*n*[TRIGEN]).

BCTU writes 0 to a software trigger field on completion of the triggered conversion.

Access: User read/write; this register is write protected by Write Protection (WRPROT). If protection is enabled, writes to this register generate a transfer error and do not modify its content.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SFTRG63	SFTRG62	SFTRG61	SFTRG60	SFTRG59	SFTRG58	SFTRG57	SFTRG56	SFTRG55	SFTRG54	SFTRG53	SFTRG52	SFTRG51	SFTRG50	SFTRG49	SFTRG48
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SFTRG47	SFTRG46	SFTRG45	SFTRG44	SFTRG43	SFTRG42	SFTRG41	SFTRG40	SFTRG39	SFTRG38	SFTRG37	SFTRG36	SFTRG35	SFTRG34	SFTRG33	SFTRG32
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SFTRGm	Software Trigger m Starts an ADC conversion. 0b - No effect 1b - Trigger conversion

61.14.8 Software Trigger 3 (SFTRGR3)

Offset

Register	Offset
SFTRGR3	234h

Function

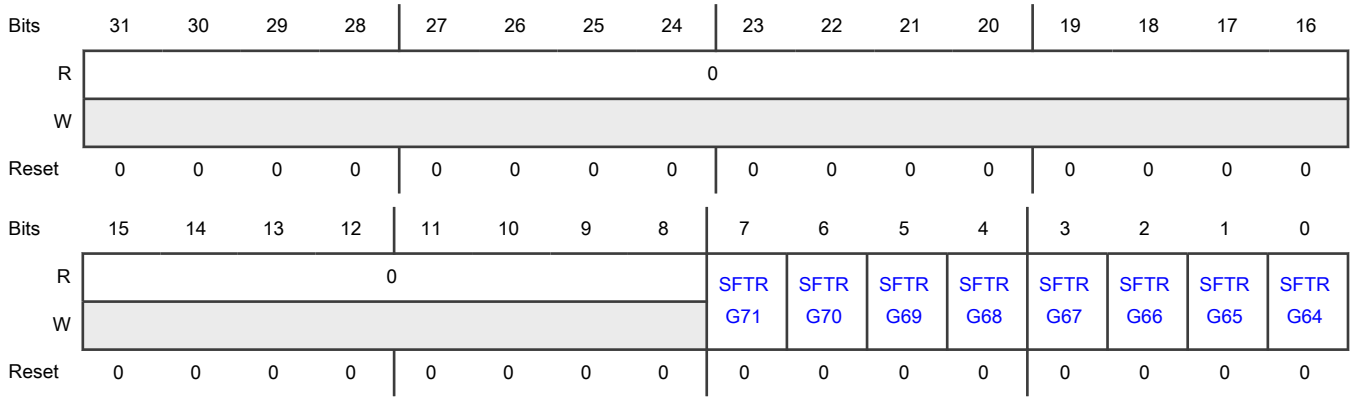
Allows you to trigger conversions on the associated triggers.

To use a software trigger, you must first disable the corresponding hardware trigger (TRGCFG_*n*[TRIGEN]).

BCTU writes 0 to a software trigger field on completion of the triggered conversion.

Access: User read/write; this register is write protected by Write Protection (WRPROT). If protection is enabled, writes to this register generate a transfer error and do not modify its content.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SFTRGm	Software Trigger m Starts an ADC conversion. 0b - No effect 1b - Trigger conversion

61.14.9 ADCn Result Data (ADC0DR - ADC2DR)

Offset

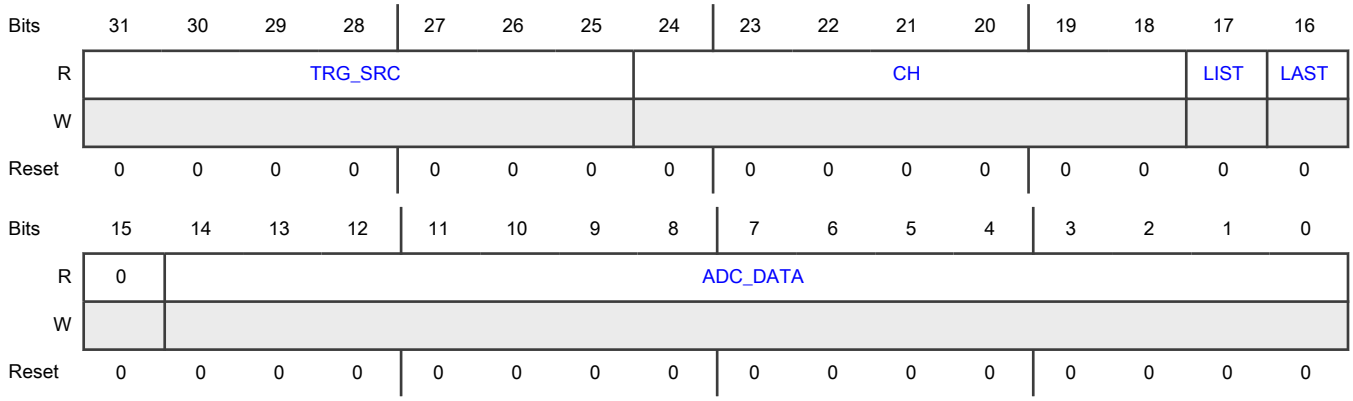
Register	Offset
ADC0DR	23Ch
ADC1DR	240h
ADC2DR	244h

Function

Provides ADC conversion result data. Also indicates the ADC channel used in that conversion, the trigger source that initiated it, and whether conversion was initiated from a CL.

Access: User read

Diagram



Fields

Field	Function
31-25 TRG_SRC	Trigger Source Contains the trigger number used to trigger the conversion.
24-18 CH	Channel Contains the ADC channel used for the conversion.
17 LIST	List Indicates whether the conversion was part of a CL or a single conversion trigger. 0b - Single conversion 1b - CL
16 LAST	Last For CL conversions, indicates this conversion result is the last conversion of the CL. 0b - Not the last conversion of a CL, or not a CL conversion 1b - Last conversion of a CL
15 —	Reserved
14-0 ADC_DATA	ADC Data Contains the data from the conversion.

61.14.10 CL Size Status (LISTSTAR)

Offset

Register	Offset
LISTSTAR	24Ch

Function

Indicates the size of the CL. Each element of the CL contains the channel number used in one conversion.

Access: User read

Diagram

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

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

Fields

Field	Function
31-8 —	Reserved
7-0 LISTSZ	CL Size Indicates the size of the CL in number of elements.

61.14.11 CL Channel Address (LISTCHR_0 - LISTCHR_15)

Offset

For m = 0 to 15:

Register	Offset
LISTCHR_m	250h + (m × 4h)

Function

Configures these attributes for two CL elements:

- ADC channel number to use for conversion
- Whether the element is the last element of a CL

Access: User read/write

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LAST_y_...	NEXT_CH...	0		0				0	ADC_CHL_y_plus_1						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LAST_y	Last Channel Specifies this element as the last channel in the CL. 0b - Not last 1b - Last channel in CL
30 NEXT_CH_WAIT_ON_TRIG_y	Next Channel Wait For Trigger Instructs CL execution to stop after executing the current ADC channel. Execution begins again when the same trigger, which started the CL, reoccurs. **NOTE** If this is the last channel in the CL (LAST_y), you must write 0 to this field. 0b - CL executes continuously 1b - CL stops executing
29-28 —	Reserved
27-24 —	Reserved
23 —	Reserved
22-16 ADC_CH_y	ADC Channel Selection Specifies the ADC channel to use for the *y* element of the CL, where *y* equals (2 x *m*).
15 LAST_y_plus_1	Last Channel Plus 1 Specifies this element as the next-to-last channel in the CL.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not next-to-last 1b - Next-to-last channel in CL
14 NEXT_CH_WAIT_ON_TRIG_y_plus_1	Next Channel Wait For Trigger Plus 1 Instructs CL execution to stop after executing the current ADC channel. Execution begins again when the same trigger, which started the CL, reoccurs. **NOTE** If this is the last channel in the CL (LAST_y_plus_1), you must write 0 to this field. 0b - CL executes continuously 1b - CL stops executing
13-12 —	Reserved
11-8 —	Reserved
7 —	Reserved
6-0 ADC_CHL_y_plus_1	ADC Channel Selection Plus 1 Specifies the ADC channel to use for the (*y* + 1) element of the CL, where *y* equals (2 x *m*).

61.14.12 FIFO Result Data (FIFO1DR - FIFO2DR)

Offset

Register	Offset
FIFO1DR	450h
FIFO2DR	454h

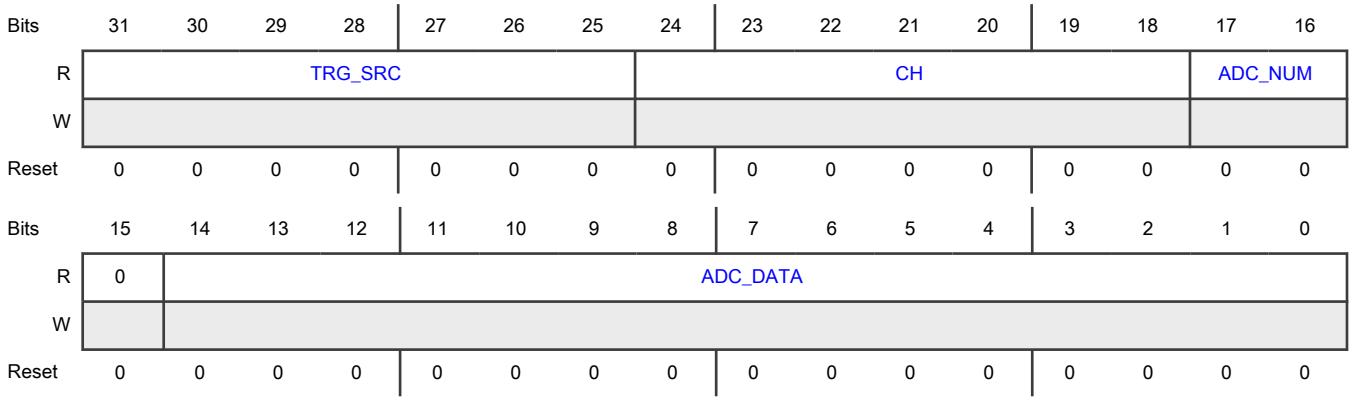
Function

Contains the ADC conversion result and information about the conversion:

- ADC channel
- Trigger source
- ADC number

Access: User read

Diagram



Fields

Field	Function
31-25 TRG_SRC	Trigger Source Indicates the trigger number used to trigger the conversion.
24-18 CH	Channel Indicates the ADC channel used for the conversion.
17-16 ADC_NUM	ADC Number Indicates the ADC used for conversion. 00b - ADC 0 01b - ADC 1 10b - ADC 2 11b - ADC 3
15 —	Reserved
14-0 ADC_DATA	ADC Data Contains the conversion data.

61.14.13 FIFO Control (FIFOCR)

Offset

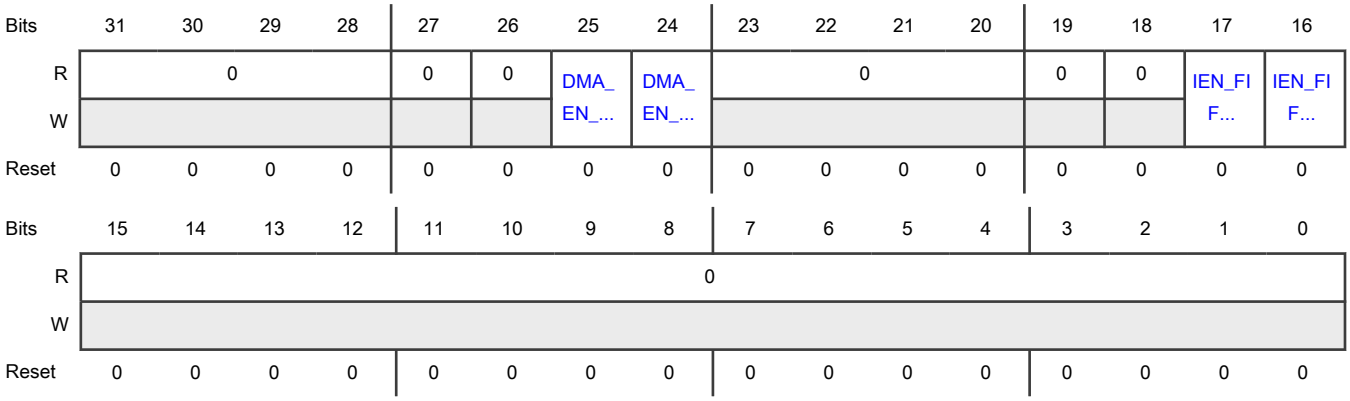
Register	Offset
FIFOCR	460h

Function

Provides control of FIFO-specific DMA, interrupts, and error interrupts.

Access: User read/write

Diagram



Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25-24 DMA_EN_FIFOn	FIFOn DMA Enable Enables a DMA request when the number of valid FIFO entries is greater than the watermark level for that FIFO. 0b - Disable 1b - Enable
23-20 —	Reserved
19 —	Reserved
18 —	Reserved
17-16 IEN_FIFOn	FIFOn Interrupt Enable Enables an interrupt request if the number of valid FIFO entries is greater than the watermark level for the FIFO.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
15-0 —	Reserved

61.14.14 FIFO Watermark Configuration (FIFOWM)

Offset

Register	Offset
FIFOWM	464h

Function

Specifies the FIFO watermark levels. If the number of active FIFO entries exceeds the watermark level, a DMA or interrupt request can be generated. See FIFO Control (FIFOCR).

Access: User read/write

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				WM_FIFO2				0				WM_FIFO1			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-11 —	Reserved
10-8 WM_FIFO2	FIFO2 Watermark Level Specifies the watermark level for FIFO2.
7-4 —	Reserved
3-0 WM_FIFO1	FIFO1 Watermark Level Specifies the watermark level for FIFO1.

61.14.15 FIFO Error/Status (FIFOERR)

Offset

Register	Offset
FIFOERR	468h

Function

Allows you to clear FIFO error and status flags. Each FIFO has its own error and status flags that can be configured to generate an interrupt. FIFO Status (FIFOSR) maintains the status of these flags. BCTU writes a 1 to these flags, and you must then clear them using FIFOERR.

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0		UNDR_ER...	OVR_ERR...	UNDR_ER...	OVR_ERR...	0				0		WM_INT_...	WM_INT_...
W					W1C	W1C	W1C	W1C							W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27 UNDR_ERR_FIFO2	Underrun Error Flag Indicates you have attempted to read from an empty FIFO. 0b - No underrun 1b - Underrun
26 OVR_ERR_FIFO2	Overrun Error Flag Indicates you have attempted to write to a full FIFO. 0b - No overrun 1b - Overrun
25 UNDR_ERR_FIFO1	Underrun Error Flag Indicates you have attempted to read from an empty FIFO. 0b - No underrun 1b - Underrun
24 OVR_ERR_FIFO1	Overrun Error Flag Indicates you have attempted to write to a full FIFO. 0b - No overrun 1b - Overrun
23-20 —	Reserved
19-18 —	Reserved
17-16 WM_INT_FIFOn	FIFO Watermark Interrupt Status Indicates the number of active entries in FIFOn exceeds the watermark level. 0b - Does not exceed watermark 1b - Exceeds watermark
15-0 —	Reserved

61.14.16 FIFO Status (FIFOSR)

Offset

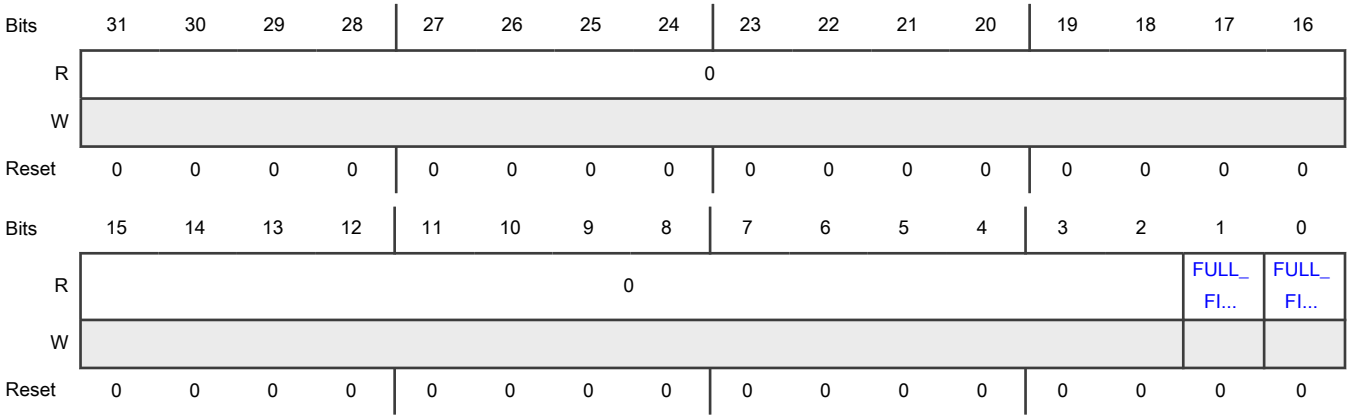
Register	Offset
FIFOSR	46Ch

Function

Contains the FIFO-full flags.

Access: User read

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 FULL_FIFOn	FIFO Full Indicates the FIFO is full. 0b - Not full 1b - Full

61.14.17 FIFO Counter (FIFOCNTR)

Offset

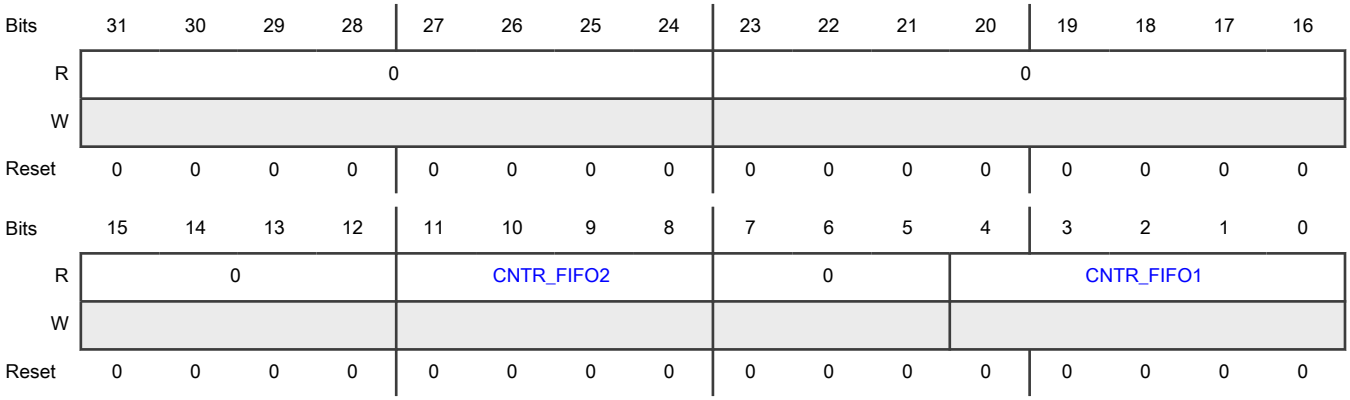
Register	Offset
FIFOCNTR	470h

Function

Indicates the number of active entries in each FIFO.

Access: User read

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-12 —	Reserved
11-8 CNTR_FIFO2	FIFO2 Counter Indicates the number of active entries in FIFO2.
7-5 —	Reserved
4-0 CNTR_FIFO1	FIFO1 Counter Indicates the number of active entries in FIFO1.

61.15 Glossary

CL Conversion list. The set of ADC channels, configured through the CL Channel Address (LISTCHR_0 - LISTCHR_15) registers, used for performing multiple sequential conversions initiated by a single trigger.

MPC Multiple parallel conversions

PSI Parallel side interface

RWB Read/write bus

WDATA Write Data bus

Chapter 62
Trigger MUX (TRGMUX)

62.1 Chip-specific TRGMUX information

62.1.1 TRGMUX input output configuration and instances

This chip has one instance of TRGMUX module.

The device supports the triggering scheme between peripherals. For the supported trigger sources and destination, see the TRGMUX connectivity file attached to this document.

This device has 16 pads (SIUL2) mapped from TRGMUX inputs and TRGMUX outputs are mapped to the eMIOS channels, hence two timers channels can use a single pin of the device to do input capture.

While using TRGMUX, below points need to be taken care:

- User must ensure the minimum pulse of 100 ns on SIUL2 pads when using them as trigger source on TRGMUX.
- Pulses which are visible on pads depends on the pad type. Different pad supports different frequencies. For more details on pad types and their respective bandwidth, see section "IO signal table" in "Signal Multiplexing" chapter.
- End of conversion (EOC) signals of ADC modules are mapped on TRGMUX inputs as trigger sources. The EOC signal is asserted after ADC conversion regardless whether conversion is signaled by polling flags, interrupt or DMA transfer. The signal shall not be used to start injected conversion on same ADC channel as it will overwrite current result register.
- The minimum pulse length requirement is 1.5X of destination clock, so that it gets properly sampled at the destination IP connected to TRGMUX outputs, otherwise there are chances of missing the triggers generated from source. For example, the trigger generated from PAD for ADC conversion should be kept high/low for at least 1.5X of ADC clock so that it gets sampled in ADC clock domain. To ensure this, pulse strechers have been placed before some of the hardware modules mapped on TRGMUX outputs.
- Some PADS are being shared by both ADC and TRGMUX. It is recommended that trigger initiated from such PADS should not be used to trigger a conversion on the ADC channel mapped on same PAD. Failing to do this will cause congestion on same PAD.
- Out of all the pads mapped on TRGMUX, first four pads have glitch filters. For details, see the TRGMUX connectivity file attached to this document. The trigger pulse width should honour the pulse width requirement as per Glitch Filter specifications. The same signal can be observed at output pin if the pulse width of the input data signal is more than 400ns and no output signal should be observed if the pulse width of the input data signal is less than 20ns. A signal with a pulse width that is between 20ns and 400 ns should not be applied as the behavior is not guaranteed.
- Trigger outputs are grouped peripheral-wise and have a common lock bit based on TRGMUX REGx. For instance, normal_trigger, injected_trigger and external_sync of ADC_0 are grouped onto TRGMUX_REG0 and have a common lock bit.

62.1.2 Pulse strechers in TRGMUX

TRGMUX has some hardware modules on its input side running at faster clock than some of the IPs present on output side. For instance, eMIOS reload outputs running at 160 MHz can trigger LPI2C trigger input clocked at 40 MHz, in that case there is a high chance that trigger from eMIOS will be missed. Following Pulse strechers are added for the IPs on output side of TRGMUX.

Table 353. Pulse strechers in TRGMUX

Evaluation Parameter	ADC_0/1/2 external trigger to sync the start pulse	BCTU trigger 23/47/71	FlexIO trigger input_0/1/2/3[1]	LPI2C_0 Trigger input	LPSPI_0/1/2 Trigger input	CM7_0/1/2/3 RXEV
IP expects	Synchronized single cycle pulse	Synchronized pulse	IP requirement is to have 2 cycle pulse of flexio_clk	IP requirement is to have 2 cycle pulse of lpi2c_clk	IP requirement is to have 2 cycle pulse of lpspi_clk	Single cycle pulse
Frequency	CORE_CLK	CORE_CLK	CORE_CLK	AIPS_SLOW_CLK	LPSPI_0 - PLAT_AIPS_CLK LPSPI_1/2/3 - AIPS_SLOW_CLK	CORE_CLK
Inside IP	Used as combo signal	Flopped inside IP and clear after ADC conversion completed	Synchronized inside IP	Synchronized inside IP	Synchronized inside IP	-
At SoC	Since it an ASYNC signal and IP required synchronized single cycle pulse, so a pulse stretcher is added to convert pulse from any frequency domain into a signle cycle pulse of CORE_CLK	Since slow IP such as LPCMP or from PAD can also trigger BCTU and BCTU wants that trigger should clear after the ADC conversion. So a pulse stretcher is added which convert any size of pulse to a single cycle pulse of BCTU clock (CORE_CLK)	Pulse stretcher is added to convert any pulse into a two cyle pulse of FLEXIO_CLK	Pulse stretcher is added to convert any pulse into a two cyle pulse of LPI2C_CLK	Pulse stretcher is added to convert any pulse into a two cyle pulse of LPSPI_CLK	Pulse stretcher is added to convert any pulse into a single cyle pulse of CORE_CLK
Output slot at which pulse stretcher is added	2, 6, 10	24, 25, 26	64, 65, 66, 67	84	88, 92, 96	156, 157

1. These pulse stretchers are available in MWCT2D17S variant only.

NOTE

The trigger outputs which have pulse stretcher before them, there should be atleast a gap of 5 cycle of destination clock for back to back trigger.

62.1.3 Trigger monitor

A trigger monitor is added before the TRGMUX inputs ADC12_0_EOC, ADC12_1_EOC and ADC12_2_EOC. You can configure the UDR registers in SIUL2 to select which trigger needs to propagate to the above TRGMUX inputs. By default after coming out of reset, all the enables are 0. Hence no trigger will be coming to these 3 TRGMUX inputs.

NOTE

Trigger monitor is not present in MWCT2D17S.

62.2 Introduction

The trigger multiplexer (TRGMUX) module allows software to configure the trigger inputs for various peripherals.

62.3 Features

The TRGMUX module allows software to select the trigger source for peripherals. The block diagram below shows the trigger selection logic of the TRGMUX module.

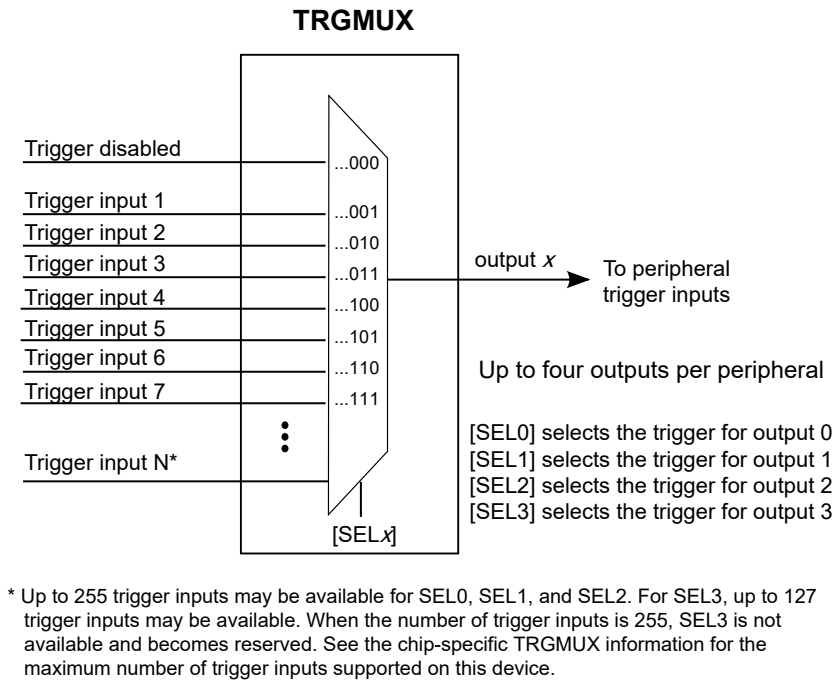


Figure 352. TRGMUX block diagram

Each peripheral has its own dedicated TRGMUX register. See each peripheral's TRGMUX register for details.

62.4 Memory map and register definition

The TRGMUX registers contain fields for selecting trigger sources for peripheral modules.

TRGMUX registers can be written only in supervisor mode.

62.4.1 TRGMUX register descriptions

62.4.1.1 TRGMUX memory map

Table 354. Select Bit Fields

Field	Description
SELx	This read/write field is used to configure the MUX select for the peripheral trigger inputs. 0000_0000 - (0x00) LOGIC 0 (VSS) 0000_0001 - (0x01) LOGIC 1 (VDD) 0000_0010 - (0x02) ADC12_0_EOC (ADC 0 end of conversion) 0000_0011 - (0x03) ADC12_1_EOC (ADC 1 end of conversion) 0000_0100 - (0x04) ADC12_2_EOC (ADC 2 end of conversion) 0000_0101 - (0x05) LPCMP_0_COUT output (LPCMP_0 comparison output (COUT)) 0000_0110 - (0x06) LPCMP_1_COUT output (LPCMP_1 comparison output (COUT)) 0000_0111 - (0x07) LPCMP_2_COUT output (LPCMP_2 comparison output (COUT)) 0000_1000 - (0x08) eDMA_eDMA_0 DONE (dma channel 0 transfer done) 0000_1001 - (0x09) eDMA_eDMA_1 DONE (dma channel 1 transfer done) 0000_1010 - (0x0A) eDMA_eDMA_16 DONE (dma channel 16 transfer done) 0000_1011 - (0x0B) eDMA_eDMA_17 DONE (dma channel 17 transfer done) 0000_1100 - (0x0C) eMIOS_0_RELOAD_OUT_CH[23] (eMIOS_0 channel 23 reload output) 0000_1101 - (0x0D) eMIOS_0_RELOAD_OUT_CH[22] (eMIOS_0 channel 22 reload output) 0000_1110 - (0x0E) eMIOS_0_RELOAD_OUT_CH[8] (eMIOS_0 channel 8 reload output) 0000_1111 - (0x0F) eMIOS_0_RELOAD_OUT_CH[0] (eMIOS_0 channel 0 reload output) 0001_0000 - (0x10) eMIOS_0_IPP_DO_eMIOS_CH[0] (eMIOS_0 channel 0 output) 0001_0001 - (0x11) eMIOS_0_IPP_DO_eMIOS_CH[1] (eMIOS_0 channel 1 output) 0001_0010 - (0x12) eMIOS_0_IPP_DO_eMIOS_CH[2] (eMIOS_0 channel 2 output) 0001_0011 - (0x13) eMIOS_0_IPP_DO_eMIOS_CH[3] (eMIOS_0 channel 3 output) 0001_0100 - (0x14) eMIOS_0_IPP_DO_eMIOS_CH[4] (eMIOS_0 channel 4 output) 0001_0101 - (0x15) eMIOS_0_IPP_DO_eMIOS_CH[5] (eMIOS_0 channel 5 output) 0001_0110 - (0x16) eMIOS_0_IPP_DO_eMIOS_CH[6] (eMIOS_0 channel 6 output) 0001_0111 - (0x17) eMIOS_0_IPP_DO_eMIOS_CH[7] (eMIOS_0 channel 7 output) 0001_1000 - (0x18) eMIOS_0_IPP_DO_eMIOS_CH[8] (eMIOS_0 channel 8 output) 0001_1001 - (0x19) eMIOS_0_IPP_DO_eMIOS_CH[9] (eMIOS_0 channel 9 output) 0001_1010 - (0x1A) eMIOS_0_IPP_DO_eMIOS_CH[10] (eMIOS_0 channel 10 output) 0001_1011 - (0x1B) eMIOS_0_IPP_DO_eMIOS_CH[11] (eMIOS_0 channel 11 output) 0001_1100 - (0x1C) eMIOS_0_IPP_DO_eMIOS_CH[12] (eMIOS_0 channel 12 output) 0001_1101 - (0x1D) eMIOS_0_IPP_DO_eMIOS_CH[13] (eMIOS_0 channel 13 output) 0001_1110 - (0x1E) eMIOS_0_IPP_DO_eMIOS_CH[14] (eMIOS_0 channel 14 output) 0001_1111 - (0x1F) eMIOS_0_IPP_DO_eMIOS_CH[15] (eMIOS_0 channel 15 output)

Table continues on the next page...

Table 354. Select Bit Fields

Field	Description
	0010_0000 - (0x20) eMIOS_0_IPP_DO_eMIOS_CH[22] (eMIOS_0 channel 22 output)
	0010_0001 - (0x21) eMIOS_0_IPP_DO_eMIOS_CH[23] (eMIOS_0 channel 23 output)
	0010_0010 - (0x22) eMIOS_1_RELOAD_OUT_CH[23] (eMIOS_1 channel 23 reload output)
	0010_0011 - (0x23) eMIOS_1_RELOAD_OUT_CH[22] (eMIOS_1 channel 22 reload output)
	0010_0100 - (0x24) eMIOS_1_RELOAD_OUT_CH[8] (eMIOS_1 channel 8 reload output)
	0010_0101 - (0x25) eMIOS_1_RELOAD_OUT_CH[0] (eMIOS_1 channel 0 reload output)
	0010_0110 - (0x26) eMIOS_1_IPP_DO_eMIOS_CH[0] (eMIOS_1 channel 0 output)
	0010_0111 - (0x27) eMIOS_1_IPP_DO_eMIOS_CH[1] (eMIOS_1 channel 1 output)
	0010_1000 - (0x28) eMIOS_1_IPP_DO_eMIOS_CH[2] (eMIOS_1 channel 2 output)
	0010_1001 - (0x29) eMIOS_1_IPP_DO_eMIOS_CH[3] (eMIOS_1 channel 3 output)
	0010_1010 - (0x2A) eMIOS_1_IPP_DO_eMIOS_CH[4] (eMIOS_1 channel 4 output)
	0010_1011 - (0x2B) eMIOS_1_IPP_DO_eMIOS_CH[5] (eMIOS_1 channel 5 output)
	0010_1100 - (0x2C) eMIOS_1_IPP_DO_eMIOS_CH[6] (eMIOS_1 channel 6 output)
	0010_1101 - (0x2D) eMIOS_1_IPP_DO_eMIOS_CH[7] (eMIOS_1 channel 7 output)
	0010_1110 - (0x2E) eMIOS_1_IPP_DO_eMIOS_CH[8] (eMIOS_1 channel 8 output)
	0010_1111 - (0x2F) eMIOS_1_IPP_DO_eMIOS_CH[9] (eMIOS_1 channel 9 output)
	0011_0000 - (0x30) eMIOS_1_IPP_DO_eMIOS_CH[10] (eMIOS_1 channel 10 output)
	0011_0001 - (0x31) eMIOS_1_IPP_DO_eMIOS_CH[11] (eMIOS_1 channel 11 output)
	0011_0010 - (0x32) eMIOS_1_IPP_DO_eMIOS_CH[12] (eMIOS_1 channel 12 output)
	0011_0011 - (0x33) eMIOS_1_IPP_DO_eMIOS_CH[13] (eMIOS_1 channel 13 output)
	0011_0100 - (0x34) eMIOS_1_IPP_DO_eMIOS_CH[14] (eMIOS_1 channel 14 output)
	0011_0101 - (0x35) eMIOS_1_IPP_DO_eMIOS_CH[15] (eMIOS_1 channel 15 output)
	0011_0110 - (0x36) eMIOS_1_IPP_DO_eMIOS_CH[22] (eMIOS_1 channel 22 output)
	0011_0111 - (0x37) eMIOS_1_IPP_DO_eMIOS_CH[23] (eMIOS_1 channel 23 output)
	0011_1000 - (0x38) FlexIO_External Output Trigger 0
	0011_1001 - (0x39) FlexIO_External Output Trigger 1
	0011_1010 - (0x3A) FlexIO_External Output Trigger 2
	0011_1011 - (0x3B) FlexIO_External Output Trigger 3
	0011_1100 - (0x3C) SIUL_TRGMUX_IN0 (Trigger input to the device through GPIO pins (TRGMUX_IN0))
	0011_1101 - (0x3D) SIUL_TRGMUX_IN1 (Trigger input to the device through GPIO pins (TRGMUX_IN1))
	0011_1110 - (0x3E) SIUL_TRGMUX_IN2 (Trigger input to the device through GPIO pins (TRGMUX_IN2))
	0011_1111 - (0x3F) SIUL_TRGMUX_IN3 (Trigger input to the device through GPIO pins (TRGMUX_IN3))
	0100_0000 - (0x40) SIUL_TRGMUX_IN4 (Trigger input to the device through GPIO pins (TRGMUX_IN4))
	0100_0001 - (0x41) SIUL_TRGMUX_IN5 (Trigger input to the device through GPIO pins (TRGMUX_IN5))

Table continues on the next page...

Table 354. Select Bit Fields

Field	Description
	0100_0010 - (0x42) SIUL_TRGMUX_IN6 (Trigger input to the device through GPIO pins (TRGMUX_IN6))
	0100_0011 - (0x43) SIUL_TRGMUX_IN7 (Trigger input to the device through GPIO pins (TRGMUX_IN7))
	0100_0100 - (0x44) SIUL_TRGMUX_IN8 (Trigger input to the device through GPIO pins (TRGMUX_IN8))
	0100_0101 - (0x45) SIUL_TRGMUX_IN9 (Trigger input to the device through GPIO pins (TRGMUX_IN9))
	0100_0110 - (0x46) SIUL_TRGMUX_IN10 (Trigger input to the device through GPIO pins (TRGMUX_IN10))
	0100_0111 - (0x47) SIUL_TRGMUX_IN11 (Trigger input to the device through GPIO pins (TRGMUX_IN11))
	0100_1000 - (0x48) SIUL_TRGMUX_IN12 (Trigger input to the device through GPIO pins (TRGMUX_IN12))
	0100_1001 - (0x49) SIUL_TRGMUX_IN13 (Trigger input to the device through GPIO pins (TRGMUX_IN13))
	0100_1010 - (0x4A) SIUL_TRGMUX_IN14 (Trigger input to the device through GPIO pins (TRGMUX_IN14))
	0100_1011 - (0x4B) SIUL_TRGMUX_IN15 (Trigger input to the device through GPIO pins (TRGMUX_IN15))
	0100_1100 - (0x4C) LPI2C_0_Master trigger output
	0100_1101 - (0x4D) LPI2C_0_Slave trigger output
	0100_1110 - (0x4E) LPSPI_0_End of frame trigger
	0100_1111 - (0x4F) LPSPI_0_Receive data trigger
	0101_0000 - (0x50) LPSPI_1_End of frame trigger
	0101_0001 - (0x51) LPSPI_1_Receive data trigger
	0101_0010 - (0x52) LPSPI_2_End of frame trigger
	0101_0011 - (0x53) LPSPI_2_Receive data trigger
	0101_0100 - (0x54) LPUART_0_trg_txword (Transmit End of Word trigger output)
	0101_0101 - (0x55) LPUART_0_trg_rxword (Receive End of Word trigger output)
	0101_0110 - (0x56) LPUART_0_trg_rxidle (Receive Idle Detected trigger output)
	0101_0111 - (0x57) LPUART_1_trg_txword (Transmit End of Word trigger output)
	0101_1000 - (0x58) LPUART_1_trg_rxword (Receive End of Word trigger output)
	0101_1001 - (0x59) LPUART_1_trg_rxidle (Receive Idle Detected trigger output)
	0101_1010 - (0x5A) LPUART_2_trg_txword (Transmit End of Word trigger output)
	0101_1011 - (0x5B) LPUART_2_trg_rxword (Receive End of Word trigger output)
	0101_1100 - (0x5C) LPUART_2_trg_rxidle (Receive Idle Detected trigger output)
	0101_1101 - (0x5D) LCU_0_LC1_out_i1 (logic unit 0 logic cell 1 output 1)
	0101_1110 - (0x5E) LCU_0_LC1_out_i2 (logic unit 0 logic cell 1 output 2)
	0101_1111 - (0x5F) LCU_0_LC1_out_i3 (logic unit 0 logic cell 1 output 3)
	0110_0000 - (0x60) LCU_0_LC1_out_i4 (logic unit 0 logic cell 1 output 4)
	0110_0001 - (0x61) LCU_0_LC2_out_i1 (logic unit 0 logic cell 2 output 1)
	0110_0010 - (0x62) LCU_0_LC2_out_i2 (logic unit 0 logic cell 2 output 2)

Table continues on the next page...

Table 354. Select Bit Fields

Field	Description
	0110_0011 - (0x63) LCU_0_LC2_out_i3 (logic unit 0 logic cell 2 output 3) 0110_0100 - (0x64) LCU_0_LC2_out_i4 (logic unit 0 logic cell 2 output 4) 0110_0101 - (0x65) LCU_0_LC3_out_i1 (logic unit 0 logic cell 3 output 1) 0110_0110 - (0x66) LCU_0_LC3_out_i2 (logic unit 0 logic cell 3 output 2) 0110_0111 - (0x67) LCU_0_LC3_out_i3 (logic unit 0 logic cell 3 output 3 0110_1000 - (0x68) LCU_0_LC3_out_i4 (logic unit 0 logic cell 3 output 4) 0110_1001 - (0x69) LCU_1_LC1_out_i1 (logic unit 1 logic cell 1 output 1) 0110_1010 - (0x6A) LCU_1_LC1_out_i2 (logic unit 1 logic cell 1 output 2) 0110_1011 - (0x6B) LCU_1_LC1_out_i3 (logic unit 1 logic cell 1 output 3) 0110_1100 - (0x6C) LCU_1_LC1_out_i4 (logic unit 1 logic cell 1 output 4) 0110_1101 - (0x6D) LCU_1_LC2_out_i1 (logic unit 1 logic cell 2 output 1) 0110_1110 - (0x6E) LCU_1_LC2_out_i2 (logic unit 1 logic cell 2 output 2) 0110_1111 - (0x6F) LCU_1_LC2_out_i3 (logic unit 1 logic cell 2 output 3) 0111_0000 - (0x70) LCU_1_LC2_out_i4 (logic unit 1 logic cell 2 output 4) 0111_0001 - (0x71) LCU_1_LC3_out_i1 (logic unit 1 logic cell 3 output 1) 0111_0010 - (0x72) LCU_1_LC3_out_i2 (logic unit 1 logic cell 3 output 2) 0111_0011 - (0x73) LCU_1_LC3_out_i3 (logic unit 1 logic cell 3 output 3) 0111_0100 - (0x74) LCU_1_LC3_out_i4 (logic unit 1 logic cell 3 output 4) 0111_0101 - (0x75) PIT0_PIT0 CH0 (PIT 0 channel 0) 0111_0110 - (0x76) PIT0_PIT0 CH1 (PIT 0 channel 1) 0111_0111 - (0x77) PIT0_PIT0 CH2 (PIT 0 channel 2) 0111_1000 - (0x78) PIT0_PIT0 CH3 (PIT 0 channel 3) 0111_1001 - (0x79) PIT0_PIT0 CH4 RTI (PIT 0 channel 4) 0111_1010 - (0x7A) PIT1_PIT1 CH0 (PIT 1 channel 0) 0111_1011 - (0x7B) PIT1_PIT1 CH1 (PIT 1 channel 1) 0111_1100 - (0x7C) PIT1_PIT1 CH2 (PIT 1 channel 2) 0111_1101 - (0x7D) PIT1_PIT1 CH3 (PIT 1 channel 3) 0111_1110 - (0x7E) CM7_0_TXEV (Cortex-M7_0 Tx event (TXEV)) 0111_1111 - (0x7F) CM7_1_TXEV (Cortex-M7_1 Tx event (TXEV))

TRGMUX base address: 4008_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TRGMUX ADC12_0 Register (ADC12_0)	32	RW	0000_0000h
4h	TRGMUX ADC12_1 Register (ADC12_1)	32	RW	0000_0000h
8h	TRGMUX ADC12_2 Register (ADC12_2)	32	RW	0000_0000h
Ch	TRGMUX LPCMP_0 Register (LPCMP_0)	32	RW	0000_0000h
10h	TRGMUX LPCMP_1 Register (LPCMP_1)	32	RW	0000_0000h
14h	TRGMUX LPCMP_2 Register (LPCMP_2)	32	RW	0000_0000h
18h	TRGMUX BCTU Register (BCTU)	32	RW	0000_0000h
1Ch	TRGMUX eMIOS012_ODIS Register (eMIOS012_ODIS)	32	RW	0000_0000h
20h	TRGMUX eMIOS0_0 Register (eMIOS0_0)	32	RW	0000_0000h
24h	TRGMUX eMIOS0_1 Register (eMIOS0_1)	32	RW	0000_0000h
28h	TRGMUX eMIOS0_2 Register (eMIOS0_2)	32	RW	0000_0000h
2Ch	TRGMUX eMIOS0_3 Register (eMIOS0_3)	32	RW	0000_0000h
30h	TRGMUX eMIOS1_0 Register (eMIOS1_0)	32	RW	0000_0000h
34h	TRGMUX eMIOS1_1 Register (eMIOS1_1)	32	RW	0000_0000h
38h	TRGMUX eMIOS1_2 Register (eMIOS1_2)	32	RW	0000_0000h
3Ch	TRGMUX eMIOS1_3 Register (eMIOS1_3)	32	RW	0000_0000h
40h	TRGMUX FlexIO Register (FlexIO)	32	RW	0000_0000h
44h	TRGMUX SIUL_OUT0 Register (SIUL_OUT0)	32	RW	0000_0000h
48h	TRGMUX SIUL_OUT1 Register (SIUL_OUT1)	32	RW	0000_0000h
4Ch	TRGMUX SIUL_OUT2 Register (SIUL_OUT2)	32	RW	0000_0000h
50h	TRGMUX SIUL_OUT3 Register (SIUL_OUT3)	32	RW	0000_0000h
54h	TRGMUX LPI2C_0 Register (LPI2C_0)	32	RW	0000_0000h
58h	TRGMUX LPSPI_0 Register (LPSPI_0)	32	RW	0000_0000h
5Ch	TRGMUX LPSPI_1 Register (LPSPI_1)	32	RW	0000_0000h
60h	TRGMUX LPSPI_2 Register (LPSPI_2)	32	RW	0000_0000h
64h	TRGMUX LPUART_0 Register (LPUART_0)	32	RW	0000_0000h
68h	TRGMUX LPUART_1 Register (LPUART_1)	32	RW	0000_0000h
6Ch	TRGMUX LPUART_2 Register (LPUART_2)	32	RW	0000_0000h
70h	TRGMUX LPUART_3 Register (LPUART_3)	32	RW	0000_0000h
74h	TRGMUX LCU0_SYNC Register (LCU0_SYNC)	32	RW	0000_0000h
78h	TRGMUX LCU0_FORCE Register (LCU0_FORCE)	32	RW	0000_0000h
7Ch	TRGMUX LCU0_0 Register (LCU0_0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
80h	TRGMUX LCU0_1 Register (LCU0_1)	32	RW	0000_0000h
84h	TRGMUX LCU0_2 Register (LCU0_2)	32	RW	0000_0000h
88h	TRGMUX LCU1_SYNC Register (LCU1_SYNC)	32	RW	0000_0000h
8Ch	TRGMUX LCU1_FORCE Register (LCU1_FORCE)	32	RW	0000_0000h
90h	TRGMUX LCU1_0 Register (LCU1_0)	32	RW	0000_0000h
94h	TRGMUX LCU1_1 Register (LCU1_1)	32	RW	0000_0000h
98h	TRGMUX LCU1_2 Register (LCU1_2)	32	RW	0000_0000h
9Ch	TRGMUX CM7_RXEV Register (CM7_RXEV)	32	RW	0000_0000h

62.4.1.2 TRGMUX ADC12_0 Register (ADC12_0)

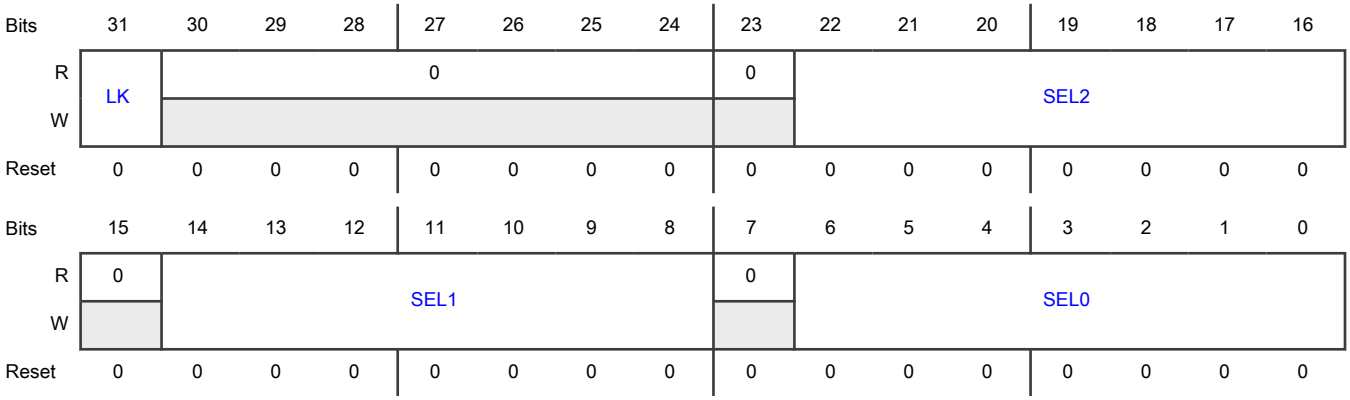
Offset

Register	Offset
ADC12_0	0h

Function

This register is for the ADC12_0 module.

Diagram



Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.3 TRGMUX ADC12_1 Register (ADC12_1)

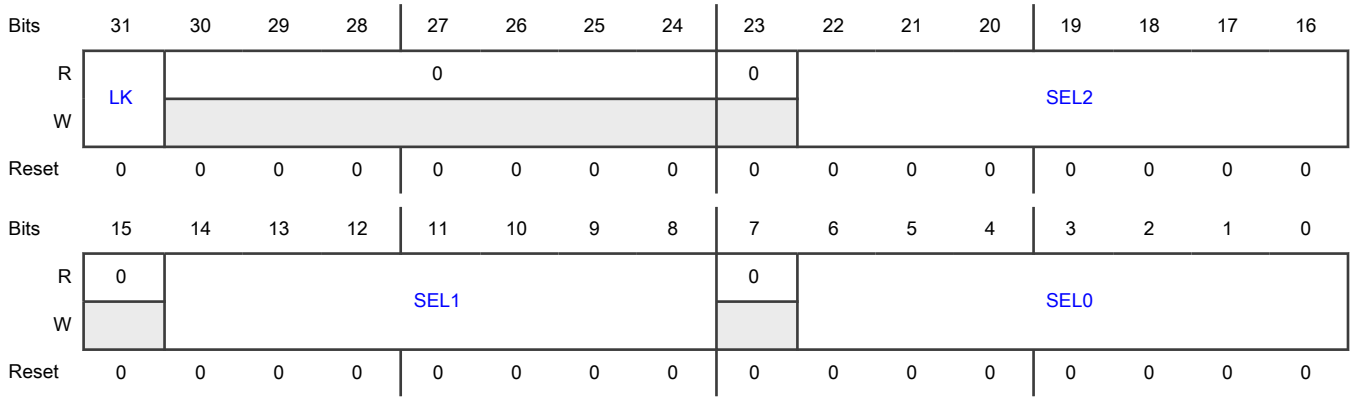
Offset

Register	Offset
ADC12_1	4h

Function

This register is for the ADC12_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.4 TRGMUX ADC12_2 Register (ADC12_2)

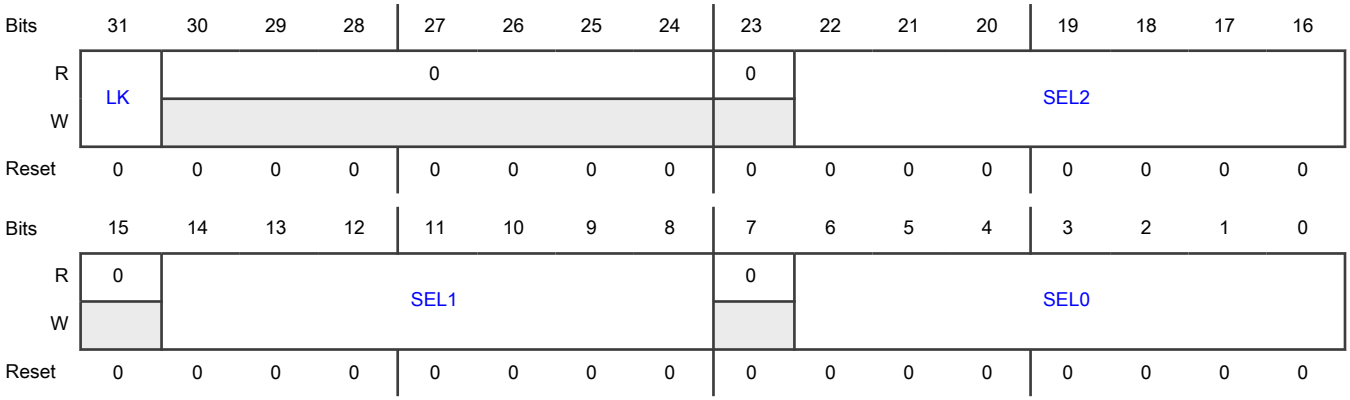
Offset

Register	Offset
ADC12_2	8h

Function

This register is for the ADC12_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.5 TRGMUX LPCMP_0 Register (LPCMP_0)

Offset

Register	Offset
LPCMP_0	Ch

Function
This register is for the LPCMP_0 module.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

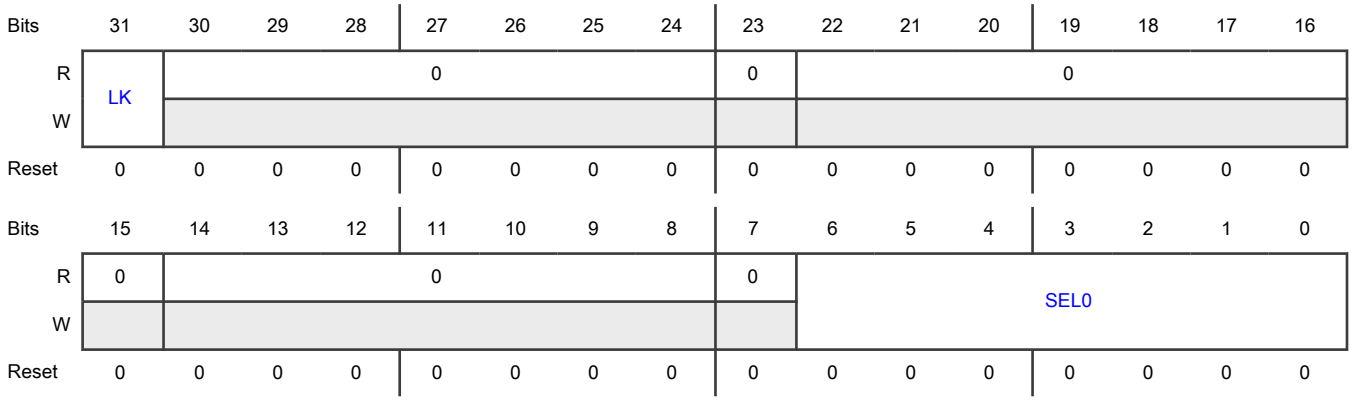
62.4.1.6 TRGMUX LPCMP_1 Register (LPCMP_1)

Offset

Register	Offset
LPCMP_1	10h

Function
This register is for the LPCMP_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

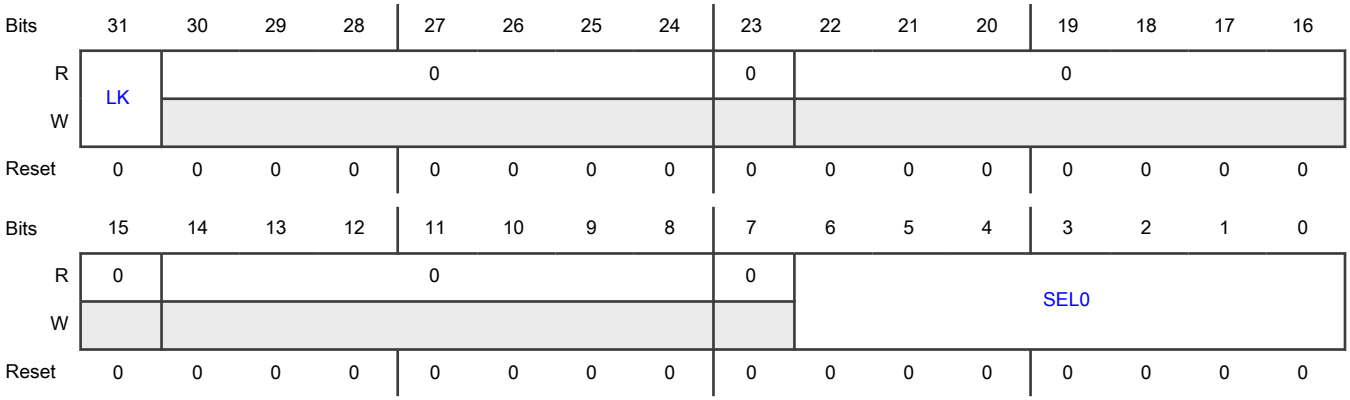
62.4.1.7 TRGMUX LPCMP_2 Register (LPCMP_2)

Offset

Register	Offset
LPCMP_2	14h

Function
This register is for the LPCMP_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.8 TRGMUX BCTU Register (BCTU)

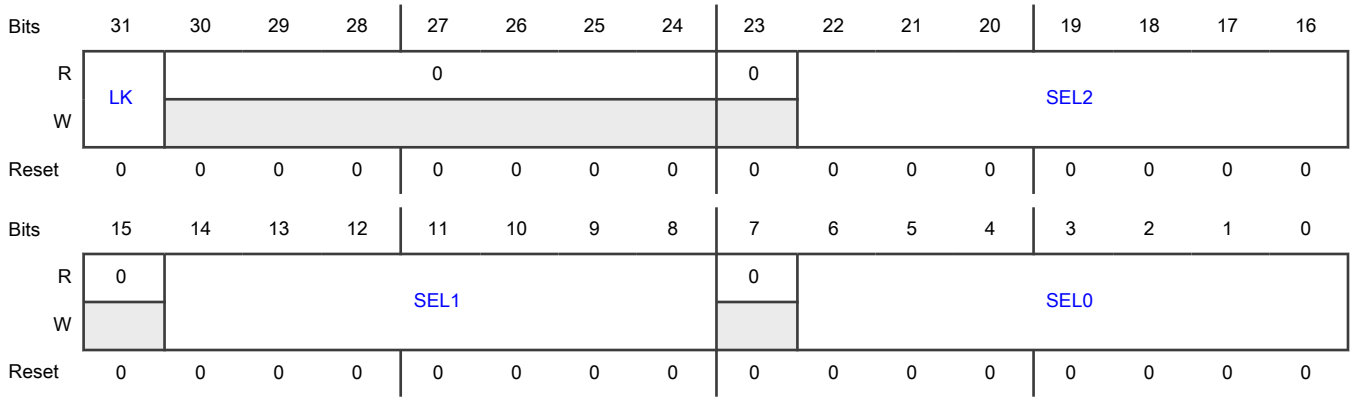
Offset

Register	Offset
BCTU	18h

Function

This register is for the BCTU module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

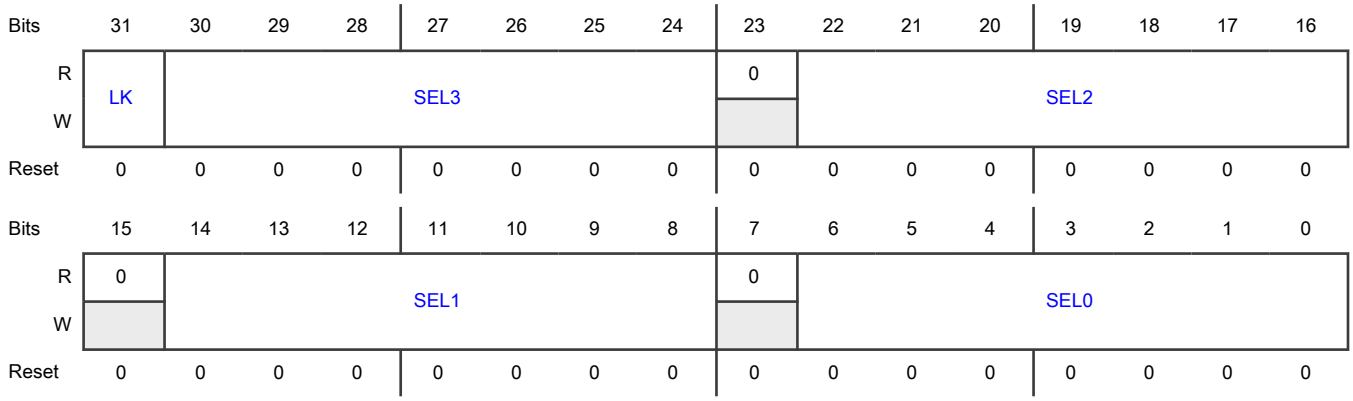
62.4.1.9 TRGMUX eMIOS012_ODIS Register (eMIOS012_ODIS)

Offset

Register	Offset
eMIOS012_ODIS	1Ch

Function
This register is for the eMIOS012_ODIS module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.10 TRGMUX eMIOS0_0 Register (eMIOS0_0)

Offset

Register	Offset
eMIOS0_0	20h

Function
This register is for the eMIOS0_0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.11 TRGMUX eMIOS0_1 Register (eMIOS0_1)

Offset

Register	Offset
eMIOS0_1	24h

Function
This register is for the eMIOS0_1 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

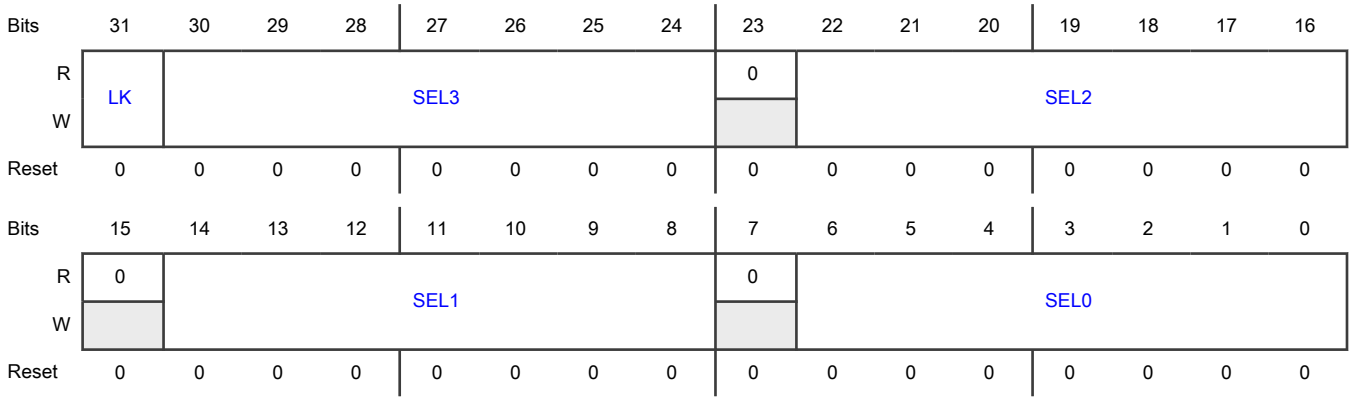
62.4.1.12 TRGMUX eMIOS0_2 Register (eMIOS0_2)

Offset

Register	Offset
eMIOS0_2	28h

Function
This register is for the eMIOS0_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

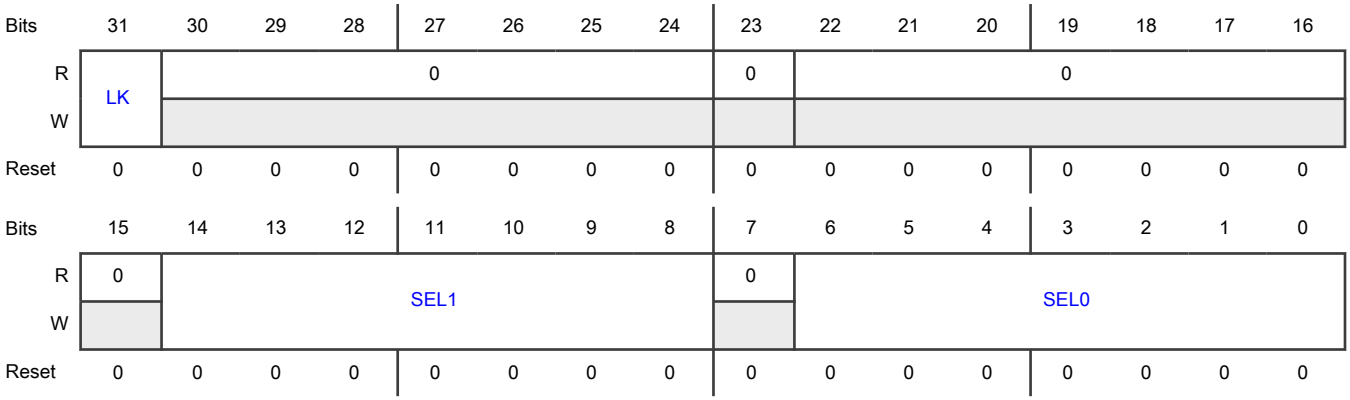
62.4.1.13 TRGMUX eMIOS0_3 Register (eMIOS0_3)

Offset

Register	Offset
eMIOS0_3	2Ch

Function
This register is for the eMIOS0_3 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.14 TRGMUX eMIOS1_0 Register (eMIOS1_0)

Offset

Register	Offset
eMIOS1_0	30h

Function
This register is for the eMIOS1_0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

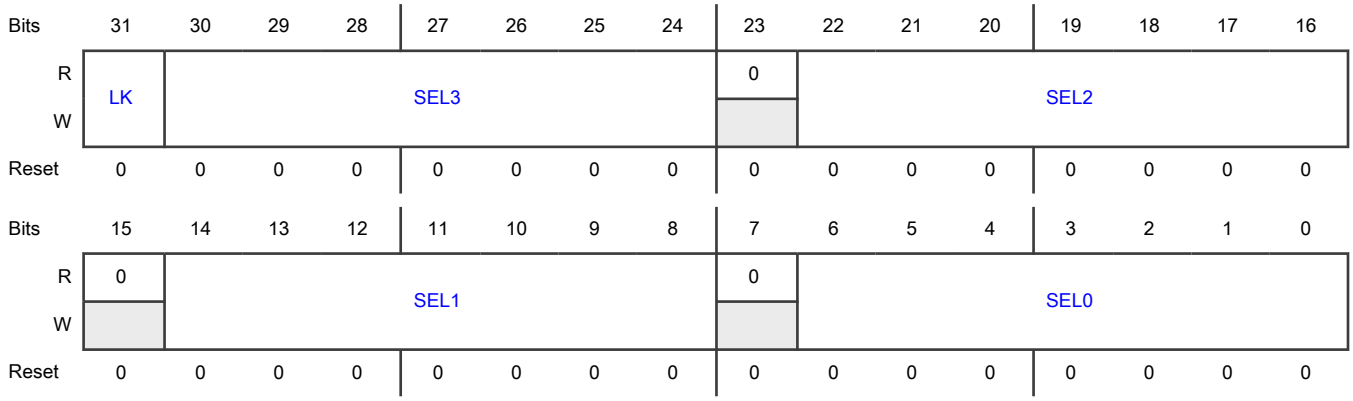
62.4.1.15 TRGMUX eMIOS1_1 Register (eMIOS1_1)

Offset

Register	Offset
eMIOS1_1	34h

Function
This register is for the eMIOS1_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

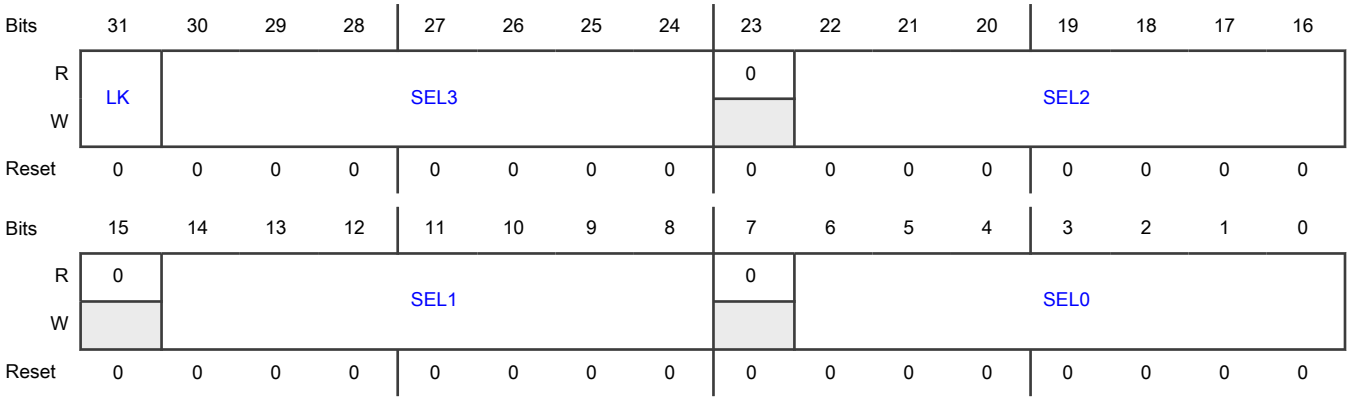
62.4.1.16 TRGMUX eMIOS1_2 Register (eMIOS1_2)

Offset

Register	Offset
eMIOS1_2	38h

Function
This register is for the eMIOS1_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.17 TRGMUX eMIOS1_3 Register (eMIOS1_3)

Offset

Register	Offset
eMIOS1_3	3Ch

Function
This register is for the eMIOS1_3 module.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.18 TRGMUX FlexIO Register (FlexIO)

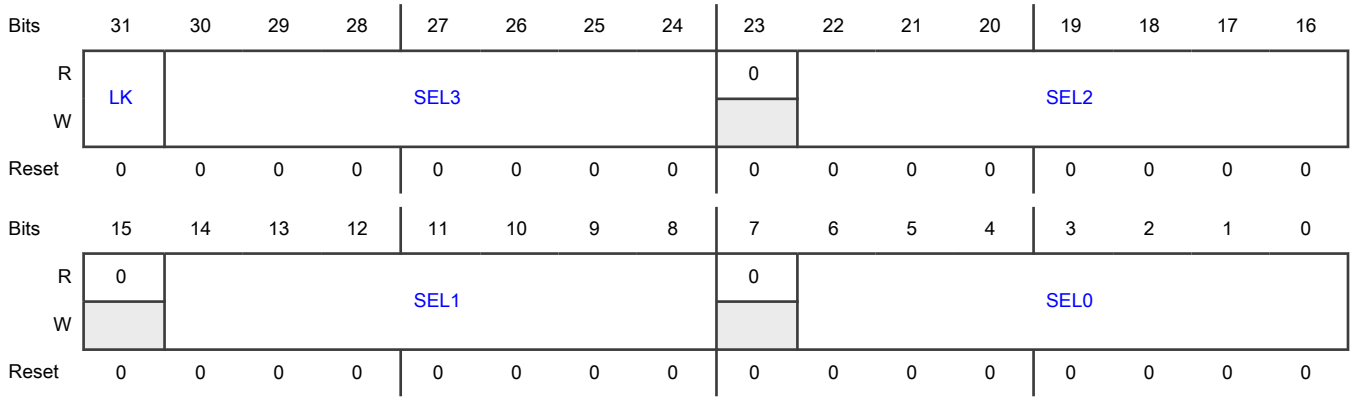
Offset

Register	Offset
FlexIO	40h

Function

This register is for the FlexIO module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

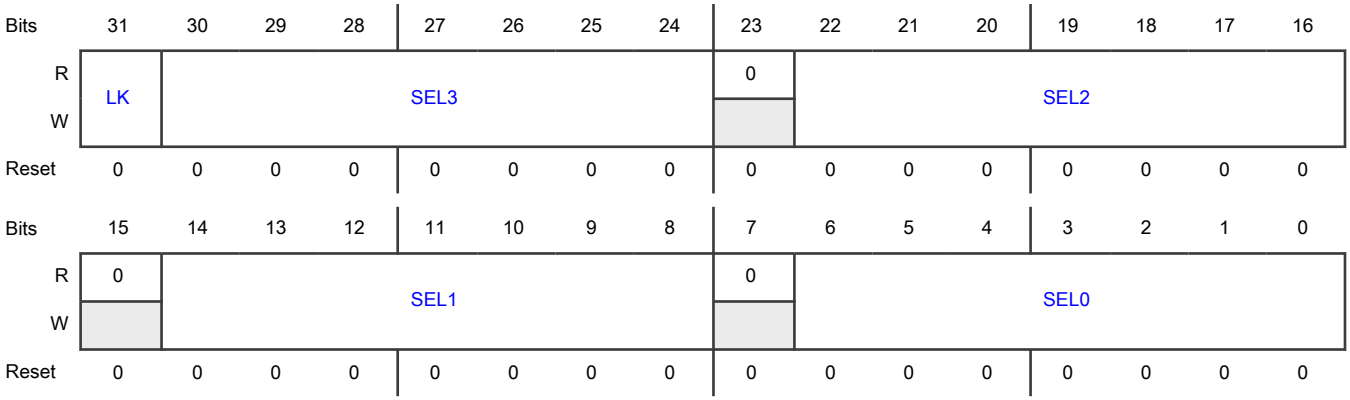
62.4.1.19 TRGMUX SIUL_OUT0 Register (SIUL_OUT0)

Offset

Register	Offset
SIUL_OUT0	44h

Function
This register is for the SIUL_OUT0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.20 TRGMUX SIUL_OUT1 Register (SIUL_OUT1)

Offset

Register	Offset
SIUL_OUT1	48h

Function
This register is for the SIUL_OUT1 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

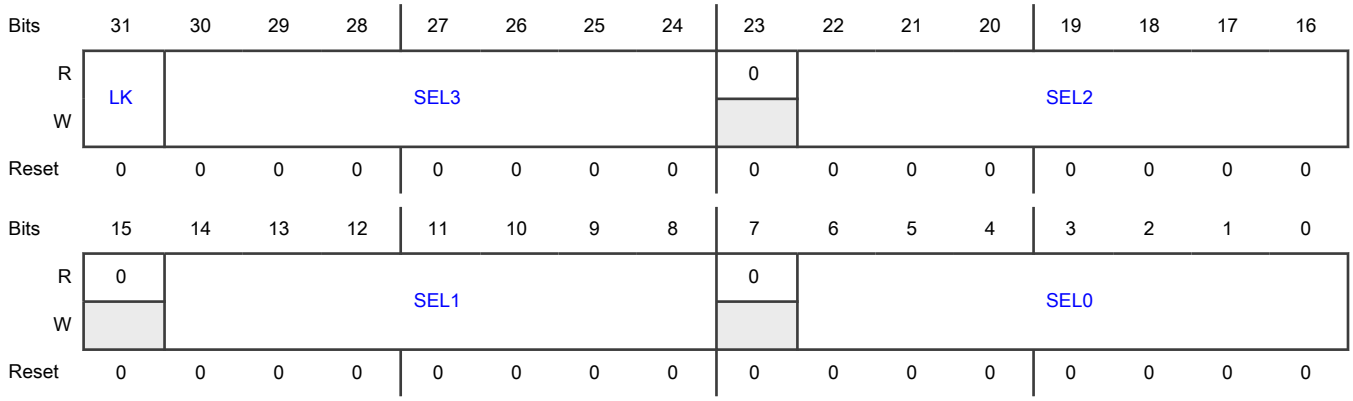
62.4.1.21 TRGMUX SIUL_OUT2 Register (SIUL_OUT2)

Offset

Register	Offset
SIUL_OUT2	4Ch

Function
This register is for the SIUL_OUT2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

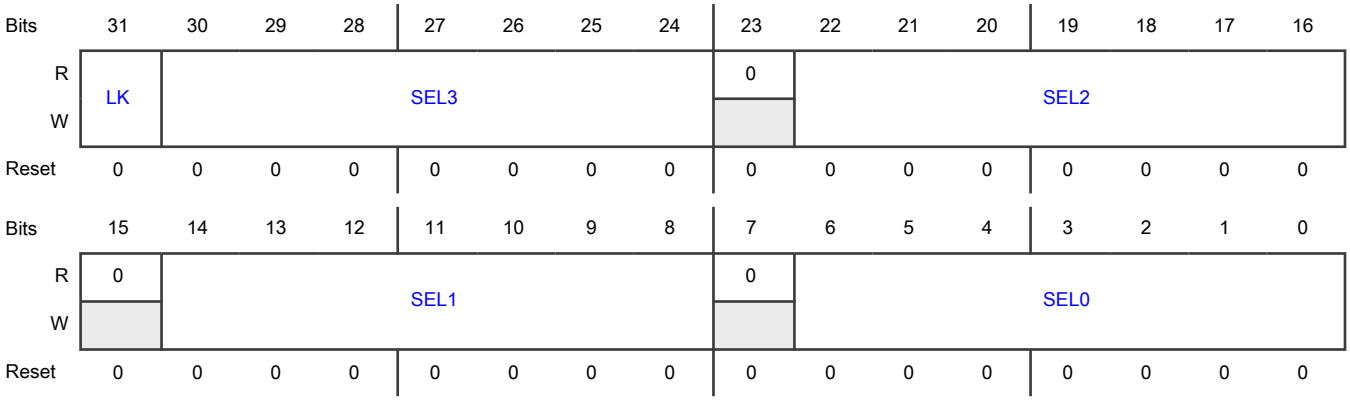
62.4.1.22 TRGMUX SIUL_OUT3 Register (SIUL_OUT3)

Offset

Register	Offset
SIUL_OUT3	50h

Function
This register is for the SIUL_OUT3 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.23 TRGMUX LPI2C_0 Register (LPI2C_0)

Offset

Register	Offset
LPI2C_0	54h

Function
This register is for the LPI2C_0 module.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

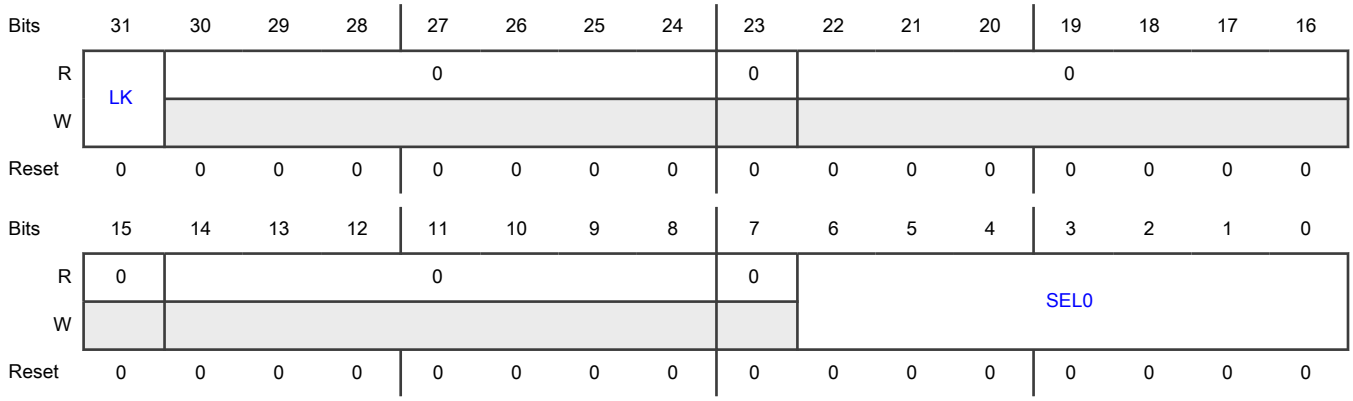
62.4.1.24 TRGMUX LPSPI_0 Register (LPSPI_0)

Offset

Register	Offset
LPSPI_0	58h

Function
This register is for the LPSPI_0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

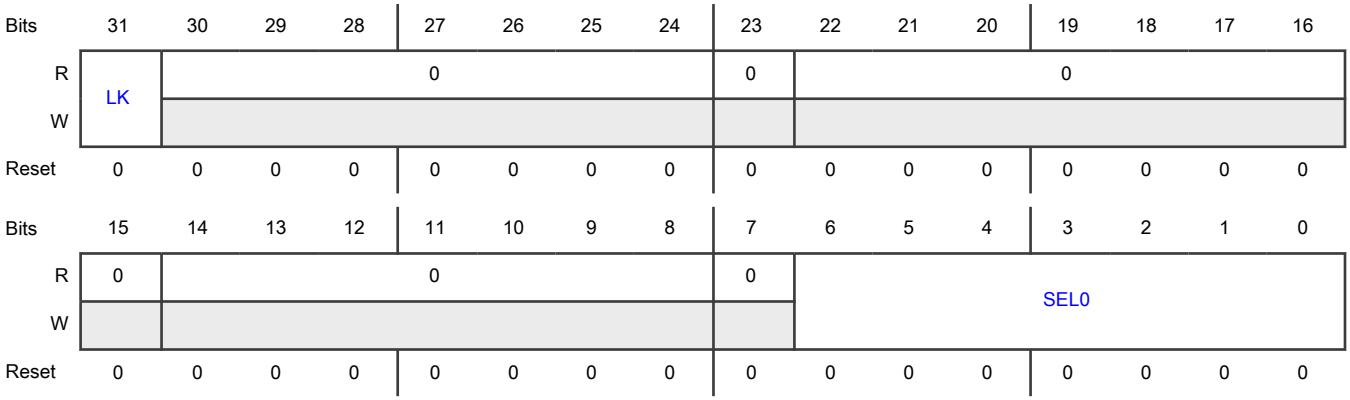
62.4.1.25 TRGMUX LPSPI_1 Register (LPSPI_1)

Offset

Register	Offset
LPSPI_1	5Ch

Function
This register is for the LPSPI_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.26 TRGMUX LPSPI_2 Register (LPSPI_2)

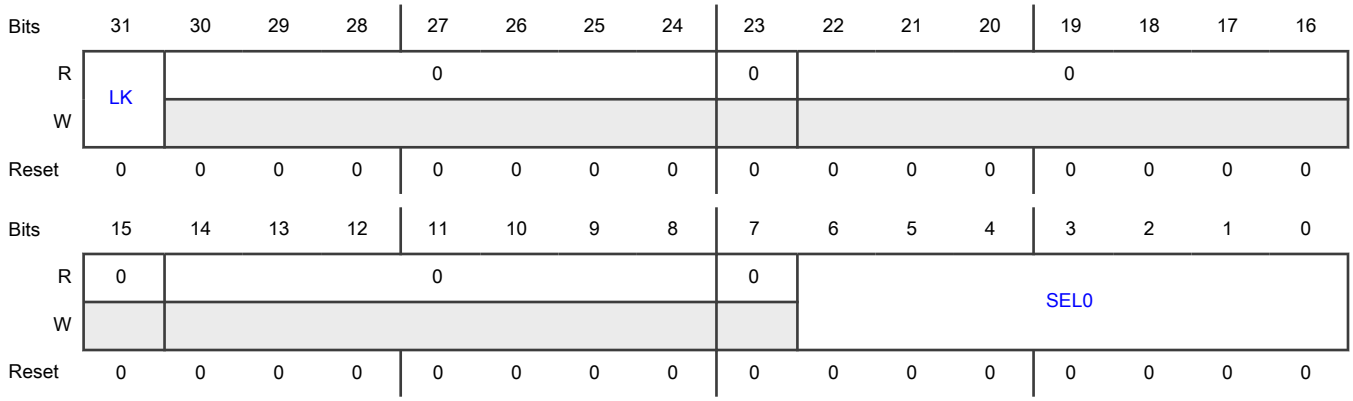
Offset

Register	Offset
LPSPI_2	60h

Function

This register is for the LPSPI_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

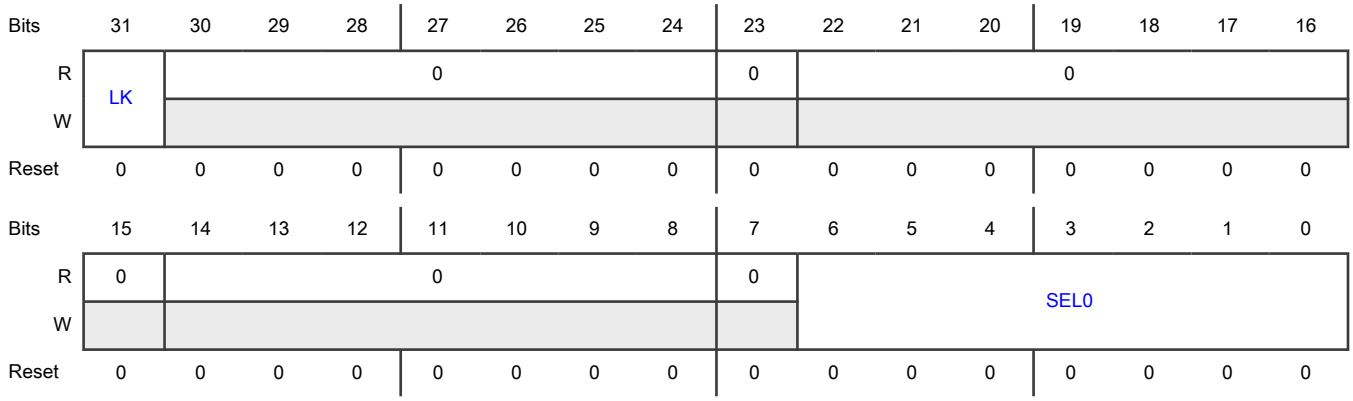
62.4.1.27 TRGMUX LPUART_0 Register (LPUART_0)

Offset

Register	Offset
LPUART_0	64h

Function
This register is for the LPUART_0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.28 TRGMUX LPUART_1 Register (LPUART_1)

Offset

Register	Offset
LPUART_1	68h

Function
This register is for the LPUART_1 module.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.29 TRGMUX LPUART_2 Register (LPUART_2)

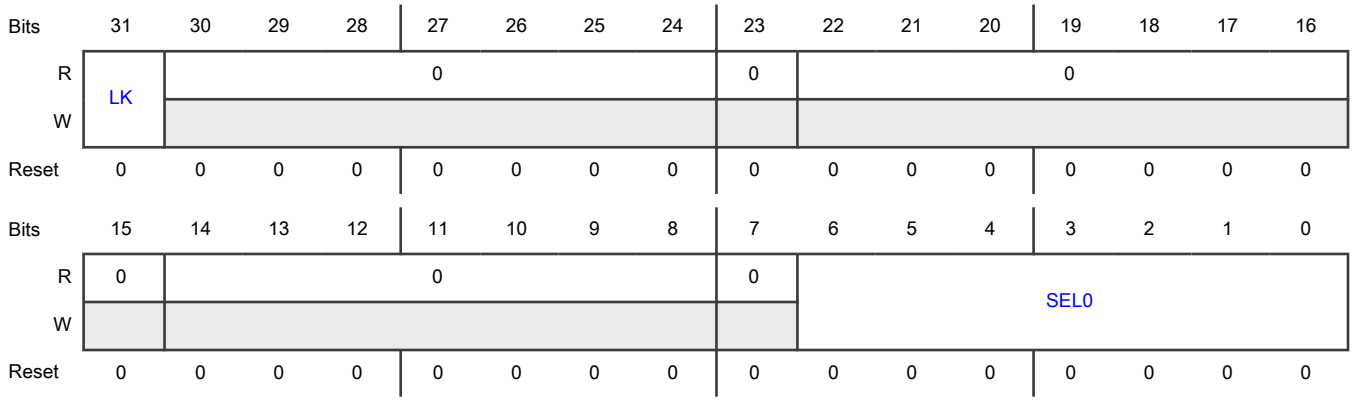
Offset

Register	Offset
LPUART_2	6Ch

Function

This register is for the LPUART_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

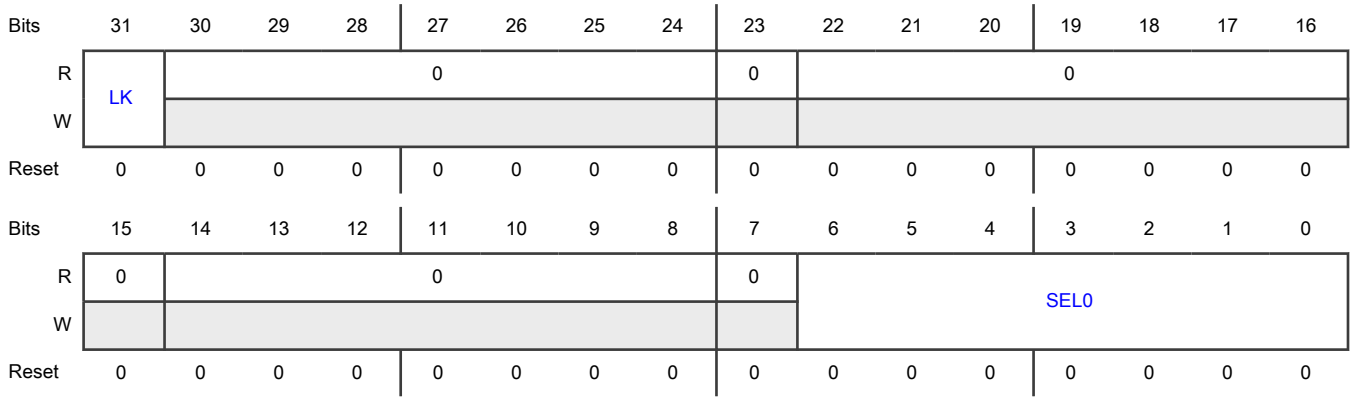
62.4.1.30 TRGMUX LPUART_3 Register (LPUART_3)

Offset

Register	Offset
LPUART_3	70h

Function
This register is for the LPUART_3 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

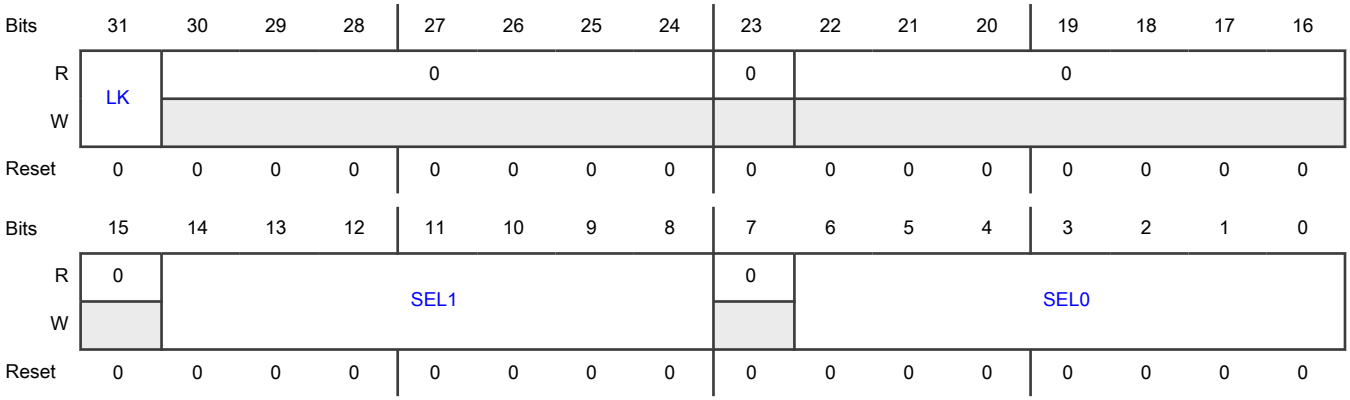
62.4.1.31 TRGMUX LCU0_SYNC Register (LCU0_SYNC)

Offset

Register	Offset
LCU0_SYNC	74h

Function
This register is for the LCU0_SYNC module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.32 TRGMUX LCU0_FORCE Register (LCU0_FORCE)

Offset

Register	Offset
LCU0_FORCE	78h

Function
This register is for the LCU0_FORCE module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.33 TRGMUX LCU0_0 Register (LCU0_0)

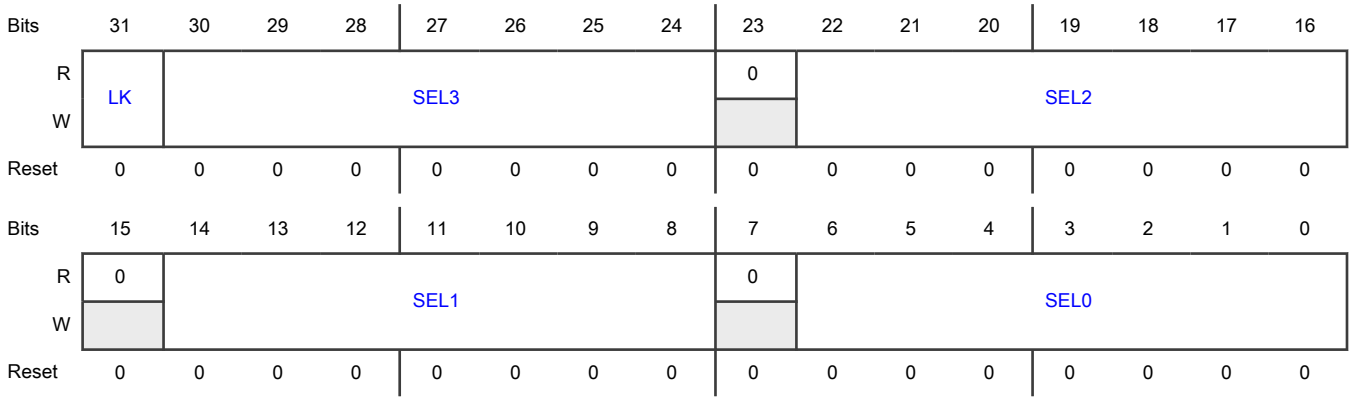
Offset

Register	Offset
LCU0_0	7Ch

Function

This register is for the LCU0_0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

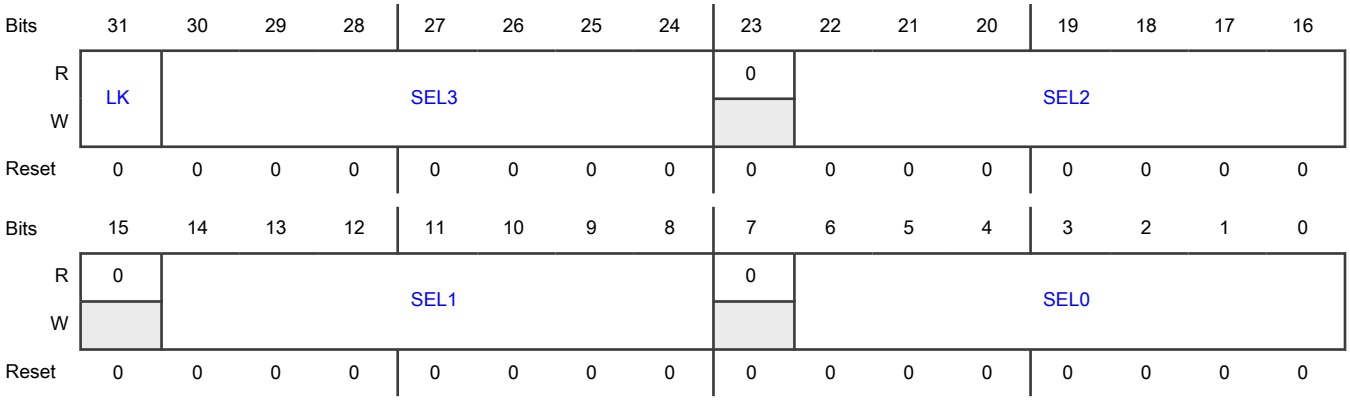
62.4.1.34 TRGMUX LCU0_1 Register (LCU0_1)

Offset

Register	Offset
LCU0_1	80h

Function
This register is for the LCU0_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.35 TRGMUX LCU0_2 Register (LCU0_2)

Offset

Register	Offset
LCU0_2	84h

Function
This register is for the LCU0_2 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

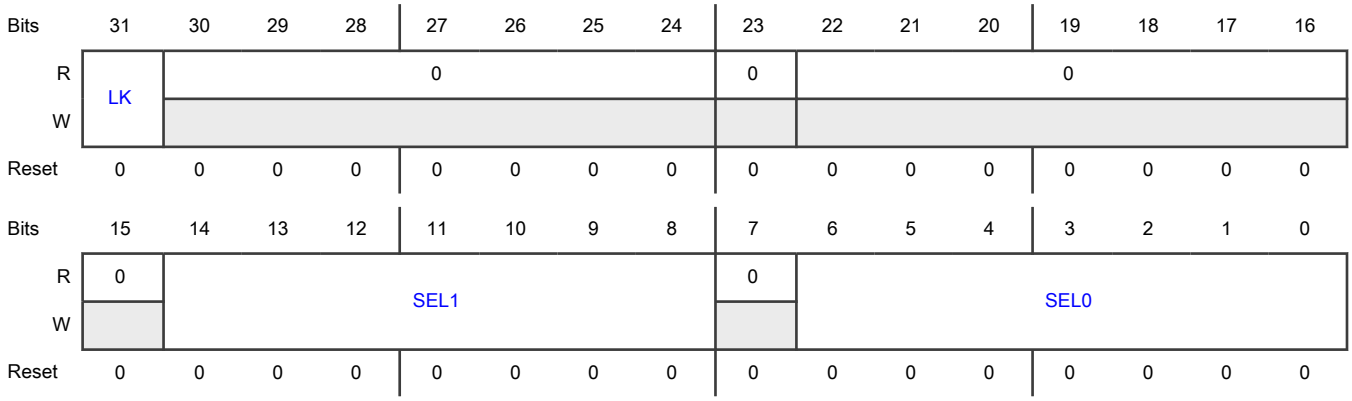
62.4.1.36 TRGMUX LCU1_SYNC Register (LCU1_SYNC)

Offset

Register	Offset
LCU1_SYNC	88h

Function
This register is for the LCU1_SYNC module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

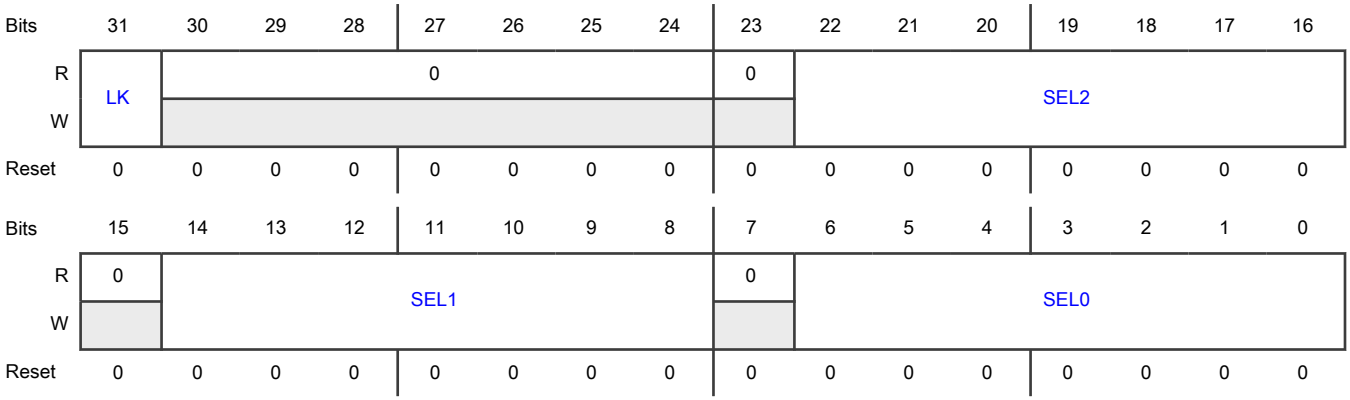
62.4.1.37 TRGMUX LCU1_FORCE Register (LCU1_FORCE)

Offset

Register	Offset
LCU1_FORCE	8Ch

Function
This register is for the LCU1_FORCE module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

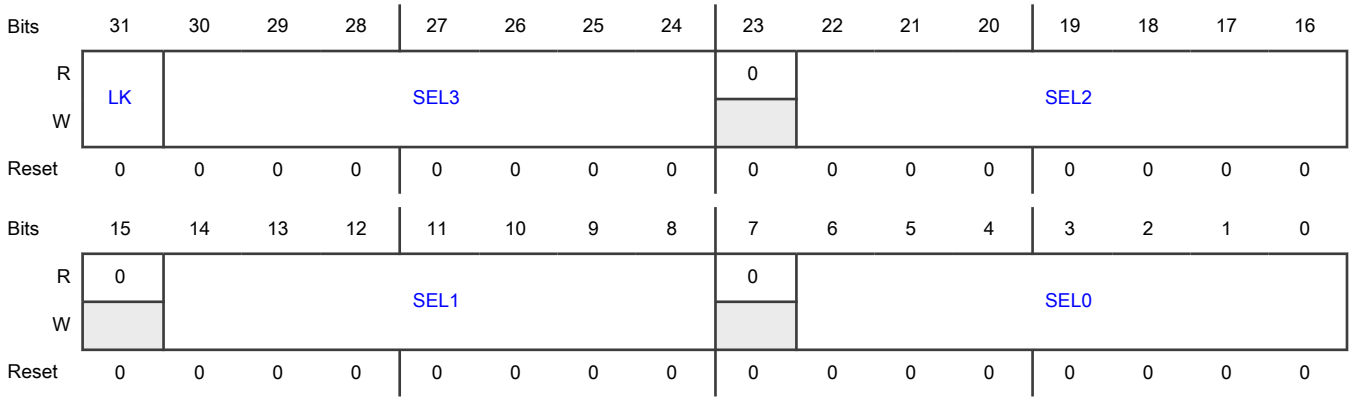
62.4.1.38 TRGMUX LCU1_0 Register (LCU1_0)

Offset

Register	Offset
LCU1_0	90h

Function
This register is for the LCU1_0 module.

Diagram



Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.39 TRGMUX LCU1_1 Register (LCU1_1)

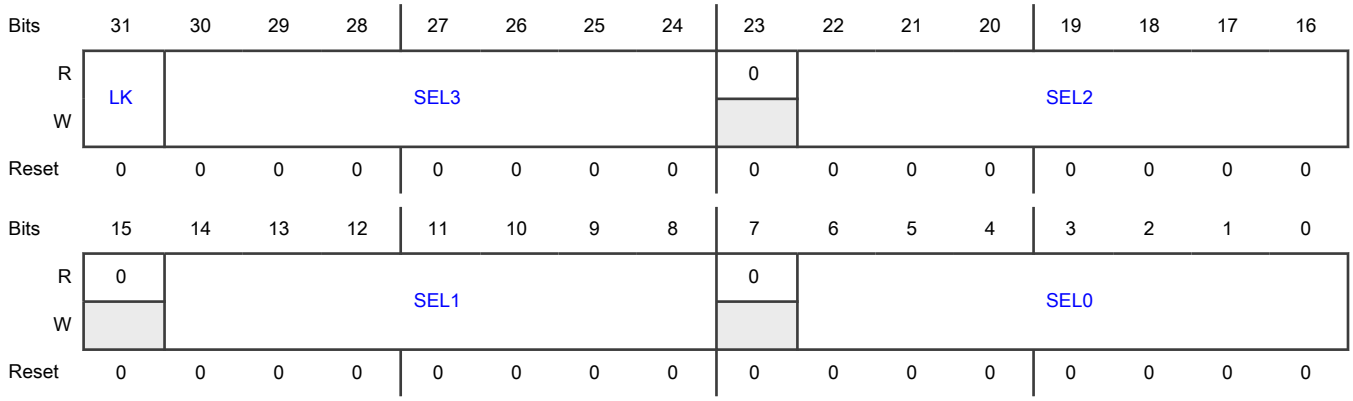
Offset

Register	Offset
LCU1_1	94h

Function

This register is for the LCU1_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

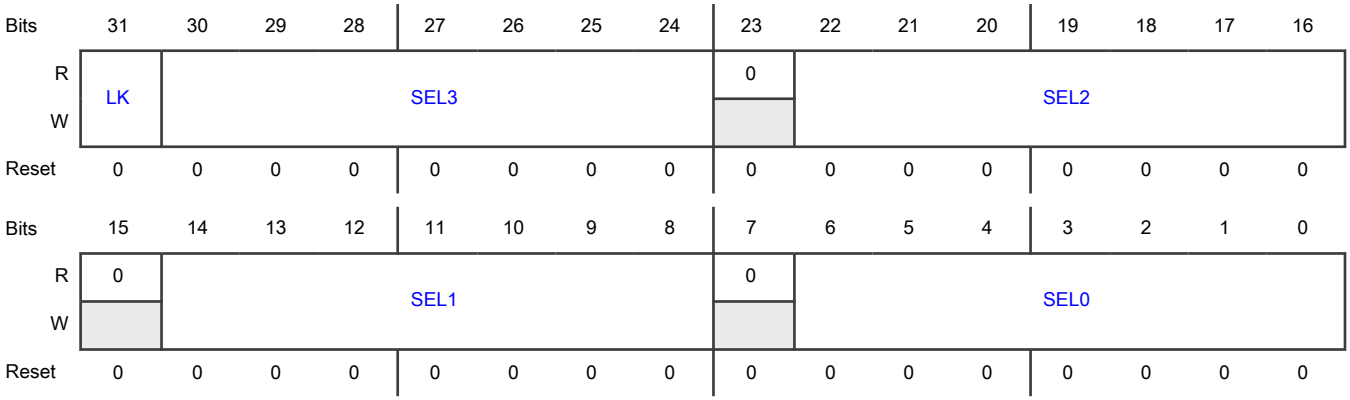
62.4.1.40 TRGMUX LCU1_2 Register (LCU1_2)

Offset

Register	Offset
LCU1_2	98h

Function
This register is for the LCU1_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

62.4.1.41 TRGMUX CM7_RXEV Register (CM7_RXEV)

Offset

Register	Offset
CM7_RXEV	9Ch

Function
This register is for the CM7_RXEV module.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

Chapter 63
Software Watchdog Timer (SWT)

63.1 Chip-specific SWT information

63.1.1 SWT instances and configuration

This chip supports up to four instances of SWT, one for each Cortex-M7 core.

Table 355. SWT instances

Instance	MWCT2D16S/MWCT2D17S	MWCT2015S/MWCT2016S
SWT_0	Yes	Yes
SWT_1	Yes	No
SWT_2	No	No
SWT_3	No	No

Table 356. SWT configuration

Instance	Core	Reset type	Clock Source	SWT_MIN_TO (100µs)	SWT_TO_RST (25ms)
SWT_0[1]	Cortex-M7_0	Functional Reset - RAM retention	SIRC (32K)	3	320h
SWT_1	Cortex-M7_1	Functional Reset - RAM retention	SIRC (32K)	3	320h
SWT_2	Cortex-M7_2	Functional Reset - RAM retention	SIRC (32K)	3	320h
SWT_3	Cortex-M7_3	Functional Reset - RAM retention	SIRC (32K)	3	320h

1. SWT_0 supports operation in STANDBY mode.

NOTE

When using SWT as a wakeup source from standby, software should ensure proper value of timeout such that SWT timeout doesn't get expired again (after wakeup till standby exit is completed). The timeout should be programmed to a value greater than the time till standby exit is complete (sum of standby exit time and the time till software disables padkeeping).

NOTE

The "Reset only the SWT" feature using SWT_RRR[RRF] is supported only when SWT reset reaction is demoted to interrupt within MC_RGM.

Protect SWT from service by non-core masters

You must program protection for the SWT registers so that only core masters can service SWT.

When an SWT timeout takes place, the logic generating a reset to the system runs off the slow internal RC oscillator (32KHz). It could therefore take the SWT almost 31.25µs to reset the system, after timeout, leaving more time for runaway code to execute. The register interface clock of SWT is running from AIPS_SLOW_CLK, see Clocking chapter for more details. The register interface clock is responsible for generating the abort for invalid access while the counter clock is responsible for generating the

SWT reset request. Since there is a synchronization happening between the register interface clock and counter clock domains before generating the SWT reset request, thus the generation of the reset request can take a number of register interface clock cycles after an invalid access is done.

63.2 Introduction

The Software Watchdog Timer (SWT) is a 32-bit window watchdog timer that enables the system to recover from situations such as:

- Software trapped in a loop
- A bus transaction failing to terminate

In regular operation, SWT requires periodic execution of a watchdog servicing operation. The servicing operation resets the timer to a specified timeout period. If this servicing action does not occur before the timer expires, SWT generates an interrupt or a hardware reset request.

You can configure SWT to generate a reset request or an interrupt on the initial timeout. SWT always generates a reset request on a second consecutive timeout.

63.3 Features

SWT has the following features:

- 32-bit countdown timer
- Selection of regular or window servicing
- Selection of reset request or interrupt on an initial timeout
- Selection of fixed or keyed service sequence
- Master access protection
- Hard and soft configuration lock bits

63.4 Initialize the SWT

To initialize the SWT, you must do the following:

- Select the clock source (if there are more than one possible sources).
- Set the timeout period.
- Control timeout behavior. There are two options:
  - Select reset request on timeout, or
  - Select interrupt on initial timeout.

You must intialize all registers before enabling SWT (CR[WEN]). You can initialize the registers in any sequence.

63.4.1 Select the clock source

See the chip-specific SWT information to find the clock source that drives the countdown timer.

63.4.2 Set the timeout period

When you enable SWT, it loads the greater of the specified watchdog timeout period or the minimum timeout period into an internal 32-bit countdown timer every time you perform a valid service operation.

To set the watchdog timeout period:

- Write the desired timeout period to TO[WTO].

The minimum timeout period is 3 clock cycles.

63.4.3 Control timeout behavior

SWT can respond to a timeout in one of two ways:

- Generate an immediate reset request on any timeout.
- Generate an interrupt and load the countdown timer on an inital timeout, then generate a reset request on a subsequent timeout.

63.4.3.1 Select reset request on timeout

To configure SWT to generate an immediate reset request on any timeout:

- Write 0 to CR[ITR].

63.4.3.2 Select interrupt on initial timeout

To configure SWT to generate an interrupt on an initial timeout:

- Write 1 to CR[ITR].

In this configuration, on the initial timeout:

- SWT loads the countdown timer with the timeout period (TO[WTO]).
- SWT indicates the interrupt with the timeout interrupt flag (IR[TIF]).
- If a second consecutive timeout occurs before writing the service sequence for the first timeout, SWT generates a reset request.

You clear the interrupt flag by writing 1 to IR[TIF].

63.4.4 Configure locking and unlocking

You can lock the SWT configuration with either a hard lock or a soft lock. When either lock is in effect, CR, TO, WN, and SK are read-only.

63.4.4.1 Enable the hard lock

To enable the hard lock:

- Write 1 to CR[HLK].

Hard lock is disabled only by a reset.

63.4.4.2 Enable the soft lock

To enable the soft lock:

- Write 1 to CR[SLK].

63.4.4.3 Unlock the soft lock

To unlock the soft lock:

1. Write C520h to SR[WSC].
2. Write D928h to SR[WSC].

This unlock sequence:

- Ignores service sequence writes.
- Recognizes the unlock sequence regardless of previous writes.
- Recognizes the unlock sequence regardless of the time between writes.
- Does not require CR[WEN] to be 1.

You can write this unlock sequence at any time.

NOTE

It is possible for a keyed service sequence to unlock soft lock. See Avoid soft unlock for the procedure to handle this situation.

63.4.4.4 Avoid soft unlock

When SWT operates in keyed sequence service, the sequence of service keys generated by the pseudorandom generator includes the unlock keys for soft lock. If one or more service routines use both unlock keys in the proper order, 0xC520 followed at any future time by 0xD928, SWT unlocks soft lock (CR[SLK] = 0). The unlock sequence logic ignores any service keys other than the unlock keys, so the unlock keys don't have to be inserted consecutively. If the second unlock key is also the second key of a service operation, unlock occurs before the service operation completes.

To avoid unlock:

If the service routine writes a key value of C520h to SR[WSC] as a key of the service operation, then do the following:

1. Complete the service operation.
2. Complete the unlock sequence by writing the second unlock key, D928h, to SR[WSC].
3. Reinitiate the soft lock (CR[SLK] = 1).

63.4.5 Select behavior on an invalid access

SWT can respond to an invalid access one of two ways:

- Generate a bus error.
- Generate a bus error and, if SWT is enabled, a reset request.

To select how SWT responds to an invalid access:

- Write the appropriate value to CR[RIA].

63.5 Enable the SWT

To enable the SWT:

- Write 1 to CR[WEN].

63.6 Initiate service operations

When enabled, SWT requires periodic execution of a servicing operation.

SWT can operate in one of the following service sequence modes:

- Fixed service sequence
- Keyed service sequence

63.6.1 Initiate a fixed service sequence

To initiate the fixed service sequence:

1. Select the fixed servicing mode (write 00b to CR[SMD]).
2. Write A602h to SR[WSC].
3. Write B480h to SR[WSC].

There is no timing requirement between the two writes. The service sequence logic ignores unlock-sequence writes.

63.6.2 Initiate a keyed service sequence

To initiate the keyed service sequence:

1. Select the keyed servicing mode (write 01b to CR[SMD]).
2. Read the initial service key from SK[SK].
3. Calculate the next service key using the provided equation.
4. Write that service key to SR[WSC].
5. Repeat steps 2 through 4 one time.

See Service key generation for information about service keys.

63.6.3 Select window service mode

In window service mode, you must write the service sequence only when the watchdog timer is less than the window start value (WN[WST]). If you write the service sequence outside of this window, the access is invalid and generates a bus error or reset request depending on the Reset on Invalid Access setting (CR[RIA]).

For example, if the timeout period is 5000 and the window start value is 1000, you must write the service sequence during the last 20% of the timeout period.

Synchronization logic in SWT causes a slight delay in the opening of the window. This delay can be up to three system clock cycles plus four counter clock cycles.

To select window service mode:

- Write 1 to (CR[WND].

63.7 Reset only the SWT

When a timeout triggers a reset request and the application does not require a system reset, you can make the SWT instance reset itself rather than the whole system.

To make a timeout reset request trigger only an SWT reset (rather than the whole system), write 1 to the Reset Request Flag (RRR[RRF]).

This action causes SWT to:

- Clear the reset request.
- If an interrupt request also occurred, clear the interrupt request (IR[TIF]).
- Reset and restart the SWT.

For systems with multiple SWT instances, this feature allows you to reset an individual SWT instance, avoiding the widespread impact of a system reset.

63.8 Test SWT operation

When you disable SWT, it loads the current countdown timer into CO[CNT]. When you enable SWT, it clears this register. You can use CO[CNT] to perform a software self-test of SWT.

To test SWT operation:

1. Enable the SWT (CR[WEN] = 1).
2. Do not service SWT for a fixed period of time that is less than the timeout value.
3. Disable the SWT (CR[WEN] = 0).
4. Read the value in CO[CNT] to determine whether the internal countdown timer is working properly.

NOTE

The value shown in CO[CNT] can lag behind the actual internal timer up to six system clock cycles plus eight counter clock cycles.

63.9 Functional description

63.9.1 Behavior in different chip modes

SWT supports the chip modes of operation as follows:

Chip mode	SWT behavior
Normal	When SWT is enabled (CR[WEN] = 1), the SWT timer counts down continuously.
Debug	If CR[FRZ] = 1, SWT stops the timer; otherwise, the timer continues to run.
Stop or Standby	If CR[STP] = 1, SWT stops the timer; otherwise, the timer continues to run.

63.9.2 Register access latency

Accesses to SWT registers occur with no peripheral bus wait states. The peripheral bus bridge may add 1 or more system wait states. However, due to synchronization logic in the SWT design, recognition of the service sequence or configuration changes may require up to three system clock cycles plus seven counter clock cycles.

63.9.3 Service key generation

SWT generates service keys with the pseudorandom key generator defined by the following equation.

$$SK_{n+1} = (17 \times SK_n + 3) \bmod 2^{16}$$

Figure 353. Pseudorandom key generator

This algorithm generates a sequence of 2^{16} different key values before repeating. In keyed service sequence mode, each time you write a valid key to SR[WSC], SWT updates SK[SK] with the next key.

For example, if the previous service key (SK[SK]) is 100h, then the next service sequence keys are 1103h followed by 2136h.

63.10 SWT register descriptions

The SWT programming model allows only 32-bit (word) accesses.

Any of the following attempted accesses are invalid:

- Non-32-bit accesses
- Writes to read-only registers
- Writes of incorrect values to SR when SWT is enabled
- Accesses to reserved addresses
- Accesses by masters without permission

You control how SWT responds to an invalid access with CR[RIA].

63.10.1 SWT memory map

SWT_0 base address: 4027_0000h

SWT_1 base address: 4046_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	FF00_010Ah
4h	Interrupt (IR)	32	RW	0000_0000h
8h	Timeout (TO)	32	RW	0000_0320h
Ch	Window (WN)	32	RW	0000_0000h
10h	Service (SR)	32	RW	0000_0000h
14h	Counter Output (CO)	32	R	0000_0000h
18h	Service Key (SK)	32	RW	0000_0000h
1Ch	Event Request (RRR)	32	RW	0000_0000h

63.10.2 Control (CR)

Offset

Register	Offset
CR	0h

Function

Contains fields for configuring and controlling SWT. The register is read-only if either hard lock or soft lock is enabled (either HLK or SLK is 1).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAP0	MAP1	MAP2	MAP3	MAP4	MAP5	MAP6	MAP7	0							
W																
Reset	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				SMD		RIA	WND	ITR	HLK	SLK	Reserved	STP	FRZ	WEN	
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0

Fields

Field	Function
31	Master Access Protection 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
MAP0	The platform bus master assignments are chip-specific. The number of this field corresponds to the XRDC DIDs of the bus master. MAP0 corresponds to XRDC DIDs 0 and 8. 0b - Access disabled 1b - Access enabled
30 MAP1	Master Access Protection 1 See the description for MAP0. MAP1 corresponds to XRDC DIDs 1 and 9. 0b - Access disabled 1b - Access enabled
29 MAP2	Master Access Protection 2 See the description for MAP0. MAP2 corresponds to XRDC DIDs 2 and 10. 0b - Access disabled 1b - Access enabled
28 MAP3	Master Access Protection 3 See the description for MAP0. MAP3 corresponds to XRDC DIDs 3 and 11. 0b - Access disabled 1b - Access enabled
27 MAP4	Master Access Protection 4 See the description for MAP0. MAP4 corresponds to XRDC DIDs 4 and 12. 0b - Access disabled 1b - Access enabled
26 MAP5	Master Access Protection 5 See the description for MAP0. MAP5 corresponds to XRDC DIDs 5 and 13. 0b - Access disabled 1b - Access enabled
25 MAP6	Master Access Protection 6 See the description for MAP0. MAP6 corresponds to XRDC DIDs 6 and 14. 0b - Access disabled 1b - Access enabled
24 MAP7	Master Access Protection 7 See the description for MAP0. MAP7 corresponds to XRDC DIDs 7 and 15.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Access disabled 1b - Access enabled
23-11 —	Reserved
10-9 SMD	Service Mode 00b - Fixed Service Sequence. To service the watchdog, write the fixed sequence 0xA602, 0xB480 to SR. 01b - Keyed Service Sequence. To service the watchdog, write two pseudorandom key values to SR. 10b - Reserved. Do not use this value. Writing this value can cause the watchdog to not be serviced. 11b - Reserved. Do not use this value. Writing this value can cause the watchdog not to be serviced.
8 RIA	Reset on Invalid Access Controls how SWT responds to an invalid access. **NOTE** For a description of how this chip implements SWT reset requests resulting from SWT invalid accesses, see the chip-specific SWT information. 0b - Generate a bus error 1b - Generate a bus error and reset request. If SWT is enabled (WEN = 1), generate a bus error and a reset request; otherwise, generate a bus error.
7 WND	Window Mode Specify regular or window mode operation. 0b - Regular mode. Can execute service sequence at any time 1b - Window mode. Can execute service sequence only when the timeout counter is less than the value in WN
6 ITR	Interrupt Then Reset Request **NOTE** For a description of how this chip implements SWT reset requests and interrupt requests resulting from SWT timeouts, see the chip-specific SWT information. 0b - Generate a reset request on a timeout 1b - Generate an interrupt on an initial timeout; generate a reset request on a second consecutive timeout
5	Hard Lock

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>HLK</td><td>Indicates that the hard lock is enabled. You cannot directly write 0 to this field. This field becomes 0 only after a reset.
0b - CR, TO, WN, and SK are read/write registers if SLK is also 0
1b - CR, TO, WN, and SK are read-only registers</td></tr>
<tr><td>4
SLK</td><td>Soft Lock
Indicates that the soft lock is enabled. You cannot directly write 0 to this field. Clear this field by writing the unlock sequence to the service register.
0b - CR, TO, WN, and SK are read/write registers if HLK is also 0
1b - CR, TO, WN, and SK are read-only registers</td></tr>
<tr><td>3
—</td><td>Reserved</td></tr>
<tr><td>2
STP</td><td>Stop Mode Control
Controls the watchdog timer when the chip enters Stop or Standby mode.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SWT_0</td><td>CR</td><td>—</td></tr><tr><td>SWT_1</td><td>—</td><td>CR</td></tr></table>
0b - Timer continues
1b - Timer stops</td></tr>
<tr><td>1
FRZ</td><td>Debug Mode Control
Controls the watchdog timer when the chip enters Debug mode.
0b - Timer continues
1b - Timer stops</td></tr>
<tr><td>0
WEN</td><td>Watchdog Enable
Enables or disables SWT.
The reset value is 0. Therefore, after reset, you must enable SWT to start the countdown timer.
0b - Disabled
1b - Enabled</td></tr>
</table>

63.10.3 Interrupt (IR)

Offset

Register	Offset
IR	4h

Function

The timeout interrupt flag.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															TIF
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 TIF	Timeout Interrupt Flag Write 1 to this field to clear the flag and interrupt. Writing a 0 has no effect. 0b - No interrupt request 1b - Interrupt request due to an initial timeout

63.10.4 Timeout (TO)

Offset

Register	Offset
TO	8h

Function

Contains the 32-bit timeout period. The register is read-only if either hard lock or soft lock is enabled (CR[HLK] or CR[SLK] is 1).

Diagram

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

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

Fields

Field	Function
31-0 WTO	Watchdog Timeout Watchdog timeout period in clock cycles. When software writes a service sequence or enables SWT, SWT loads the internal 32-bit countdown timer with this value or 3h, whichever is greater.

63.10.5 Window (WN)

Offset

Register	Offset
WN	Ch

Function

Contains the 32-bit window start value. SWT clears this register on reset. The register is read-only if either hard lock or soft lock is enabled (CR[HLK] or CR[SLK] is 1).

Diagram

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

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

Fields

Field	Function
31-0 WST	Window Start Value When you enable window mode (CR[WND]), you can write the service sequence only when the internal timer is less than this value.

63.10.6 Service (SR)

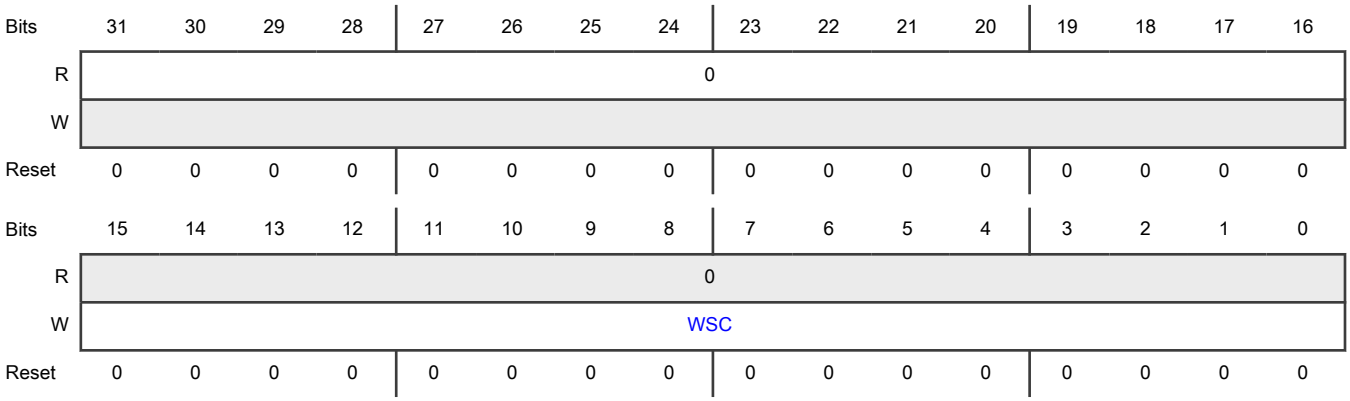
Offset

Register	Offset
SR	10h

Function

Initiates the service operation and resets the watchdog timer.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 WSC	Watchdog Service Code Use this field to service the watchdog and to unlock the Soft Lock (CR[SLK]). To service the watchdog: If SWT is in keyed service mode (CR[SMD]), write two pseudorandom key values to WSC (see Service key generation for details). Otherwise, write the following values to WSC, in the order shown:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1. A602h 2. B480h To unlock the Soft Lock (CR[SLK]), write the following values to WSC, in the order shown: 1. C520h 2. D928h When read, WSC always returns zero.

63.10.7 Counter Output (CO)

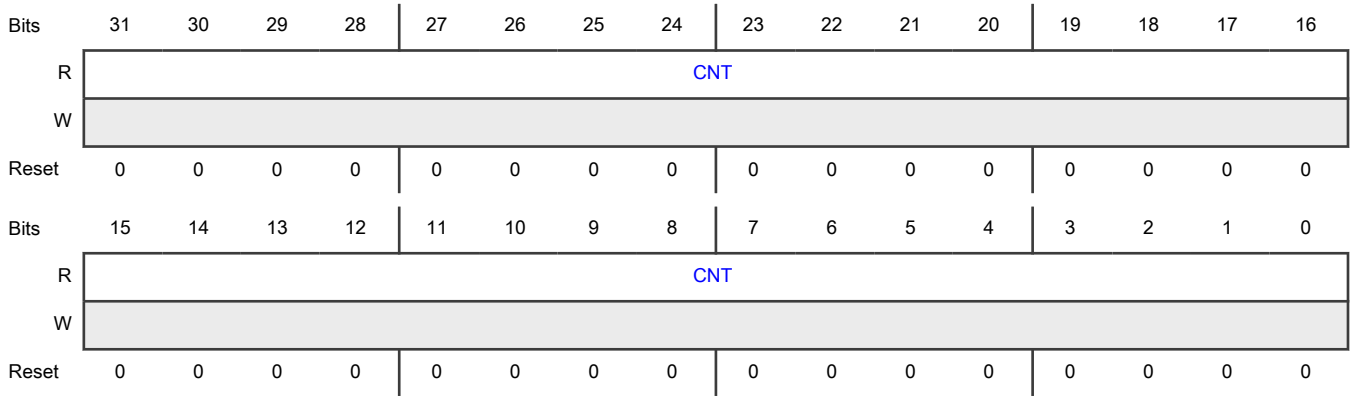
Offset

Register	Offset
CO	14h

Function

Shows the value of the internal timer when SWT is disabled.

Diagram



Fields

Field	Function
31-0 CNT	Watchdog Count When SWT is disabled (CR[WEN] is 0), CNT shows the value of the internal timer. When SWT is enabled (CR[WEN] is 1), it writes 0 to CNT. Values in this field can lag behind the internal timer value up to six system clock cycles plus eight counter clock cycles. Therefore, the CNT value that is read immediately after disabling SWT may be higher than the actual value of the internal timer.

63.10.8 Service Key (SK)

Offset

Register	Offset
SK	18h

Function

Holds the previous (or initial) service key value. This register is read-only if either hard lock or soft lock is enabled (CR[HLK] or CR[SLK] is 1).

Diagram

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

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

Fields

Field	Function
31-16 —	Reserved
15-0 SK	Service Key Holds the previous (or initial) service key value used in Initiate a keyed service sequence. If CR[SMD] is 01b, the next key value to write to SR is (17 * SK + 3) mod 2^{16}.

63.10.9 Event Request (RRR)

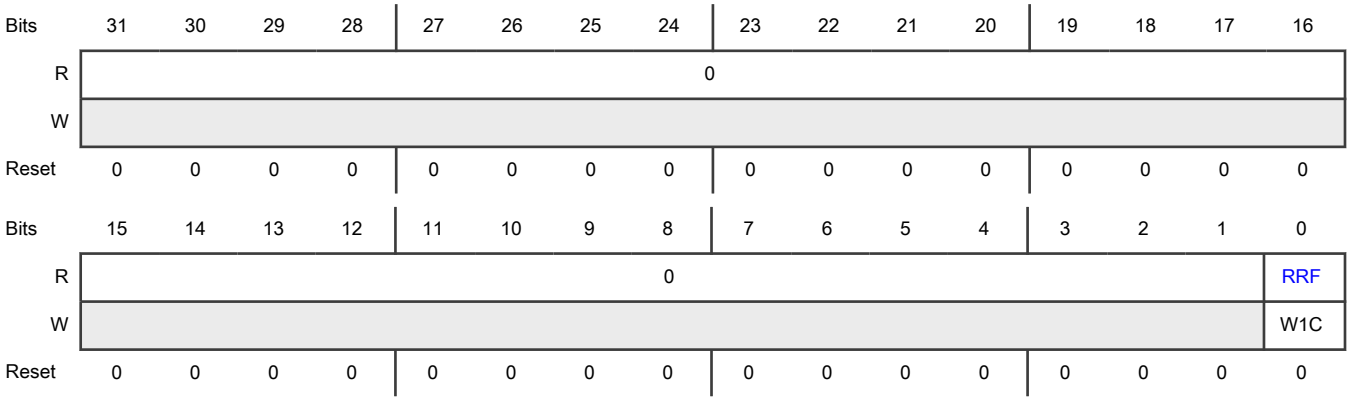
Offset

Register	Offset
RRR	1Ch

Function

Contains the timeout reset request flag. See the chip-specific information for the specific event associated with this flag.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 RRF	Reset Request Flag Write 1 to clear the flag and request. Writing 0 has no effect. 0b - No reset request 1b - Any reset request initiated

Chapter 64
System Timer Module (STM)

64.1 Chip-specific STM information

64.1.1 STM instances and configuration

This chip has up to four instances of STM, one for each Cortex-M7 core. The STM counter increments at the STM module clock frequency divided by a pre-scaled value.

Table 357. STM instances

Instance	MWCT2D16S/MWCT2D17S	MWCT2015S/MWCT2016S
STM_0	Yes	Yes
STM_1	Yes	No
STM_2	No	No
STM_3	No	No

Table 358. STM configuration

Instance	Description	Number of Channels	Number of Registers (32-bit)
STM_0	Intended for Cortex-M7_0 core	4	14
STM_1	Intended for Cortex-M7_1 core	4	14
STM_2	Intended for Cortex-M7_2 core	4	14
STM_3	Intended for Cortex-M7_3 core	4	14

64.2 Introduction

The System Timer Module (STM) supports commonly required system and application software timing functions. STM includes a 32-bit count-up timer and four 32-bit compare channels with a separate interrupt source for each channel. The timer is driven by the STM module clock divided by an 8-bit prescale value (1 to 256).

64.3 Features

STM has the following features:

- One 32-bit count-up timer with an 8-bit prescaler
- Four 32-bit compare channels
- An independent interrupt source for each channel
- Ability to stop the timer in Debug mode

64.4 Block diagram

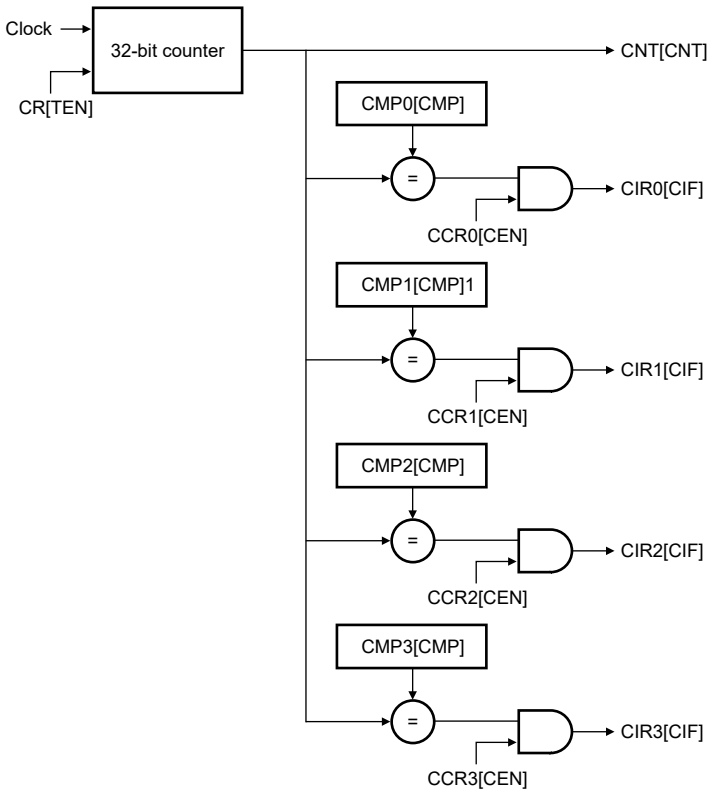


Figure 354. Block diagram

64.5 Functional description

64.5.1 Count-up timer

STM has one 32-bit count-up timer that serves as the time base for four compare channels. When enabled, the counter increments at the module clock frequency divided by a prescaler value in the range from 1 to 256. When enabled in Normal mode, the timer increments continuously. The counter rolls over at FFFF_FFFFh to 0000_0000h with no restrictions at this boundary.

64.5.2 Compare channels

STM has four identical compare channels. Each channel includes a channel control register (CCR*n*), a channel interrupt register (CCR*n*), and a channel compare register (CMP*n*). When the channel is enabled and its channel compare value matches the timer count, STM sets the channel interrupt flag and generates an IRQ on that channel.

64.5.3 Behavior in different chip modes

STM supports the chip modes of operation as follows:

Chip mode	STM behavior
Normal	When the timer is enabled (CR[TEN] = 1), the timer counts up continuously.
Debug	If CR[FRZ] = 1, STM stops the timer. Otherwise, when the timer is enabled (CR[TEN] = 1), the timer counts up continuously.

64.6 Application information

64.6.1 Configure the timer

1. Set the initial timer count (CNT[CNT]).
2. Specify STM behavior in chip Debug mode (CNT[FRZ]).
3. Set the counter prescaler (CR[CPS]).
4. Start the timer (CR[TEN]).

64.6.2 Configure the compare channels

For each compare channel:

1. Set the channel compare value (CMP*n*[CMP]).
2. Enable the compare channel (CCR*n*[CEN]).

64.6.3 Respond to compare channel events

For each compare channel:

1. Check the channel interrupt flag (CIR*n*[CIF]).
2. If the channel interrupt flag is set, respond to the interrupt request.
3. When the channel interrupt has been handled, clear the channel interrupt flag (CIR*n*[CIF]).

64.7 STM register descriptions

The STM programming model allows only 32-bit (word) accesses. An attempted reference using a different size or to a reserved address generates a bus error termination.

64.7.1 STM memory map

STM_0 base address: 4027_4000h

STM_1 base address: 4047_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	0000_0000h
4h	Count (CNT)	32	RW	0000_0000h
10h	Channel Control (CCR0)	32	RW	0000_0000h
14h	Channel Interrupt (CIR0)	32	RW	0000_0000h
18h	Channel Compare (CMP0)	32	RW	0000_0000h
20h	Channel Control (CCR1)	32	RW	0000_0000h
24h	Channel Interrupt (CIR1)	32	RW	0000_0000h
28h	Channel Compare (CMP1)	32	RW	0000_0000h
30h	Channel Control (CCR2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
34h	Channel Interrupt (CIR2)	32	RW	0000_0000h
38h	Channel Compare (CMP2)	32	RW	0000_0000h
40h	Channel Control (CCR3)	32	RW	0000_0000h
44h	Channel Interrupt (CIR3)	32	RW	0000_0000h
48h	Channel Compare (CMP3)	32	RW	0000_0000h

64.7.2 Control (CR)

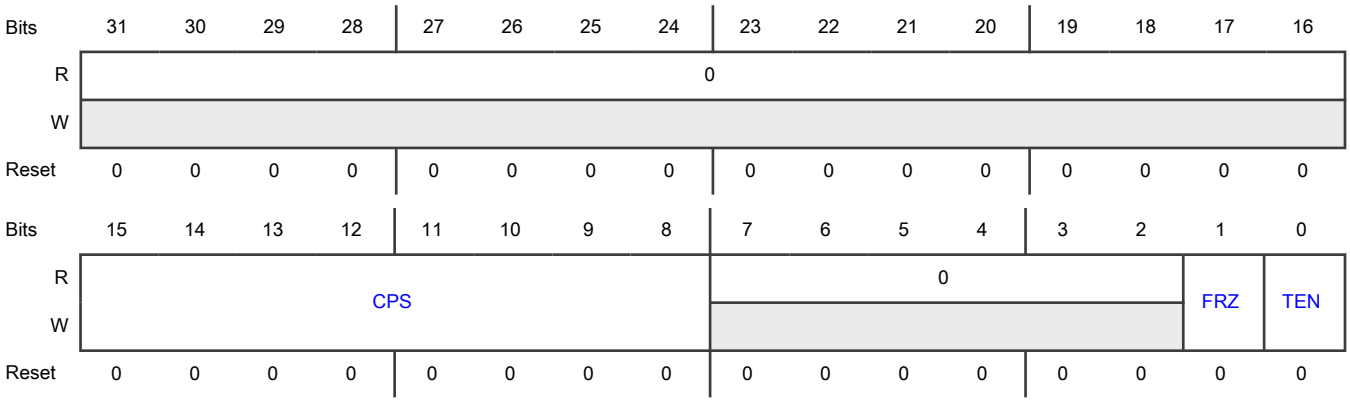
Offset

Register	Offset
CR	0h

Function

Contains fields for the prescale value, freeze control, and timer enable.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 CPS	Counter Prescaler Selects the module clock divide value for the prescaler (1–256).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 00h - Divide module clock by 1 • 01h - Divide module clock by 2 • ... • FFh - Divide module clock by 256
7-2 —	Reserved
1 FRZ	Freeze Stops the timer when the chip enters Debug mode. **NOTE** When the chip enters Debug mode, it notifies STM, which in turn uses this field to determine timer operation. 0b - Timer runs in Debug mode 1b - Timer stops in Debug mode
0 TEN	Timer Enable Enables the module timer. 0b - Disabled 1b - Enabled

64.7.3 Count (CNT)

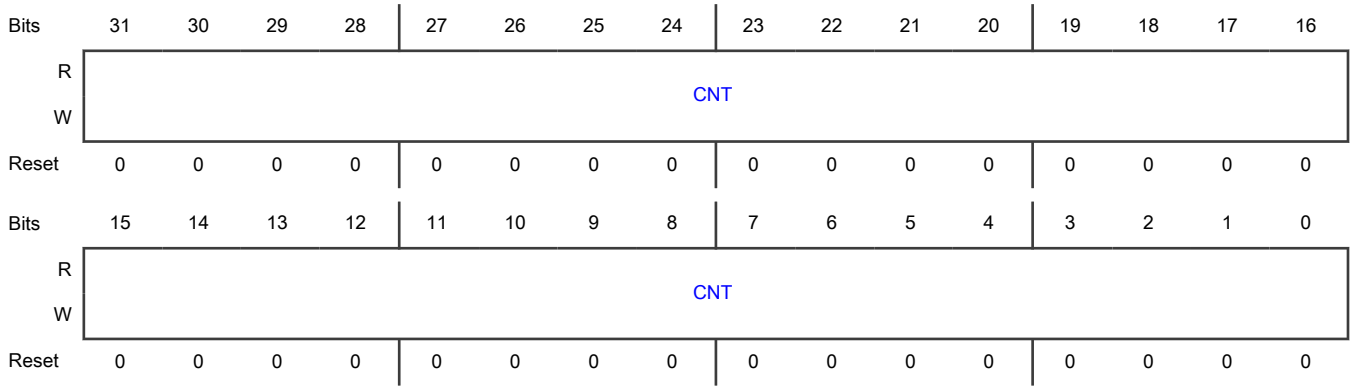
Offset

Register	Offset
CNT	4h

Function

Holds the timer count value.

Diagram



Fields

Field	Function
31-0 CNT	Timer Count The time base for all compare channels. When enabled, the timer count increments at the rate of the module clock divided by the prescale value.

64.7.4 Channel Control (CCR0 - CCR3)

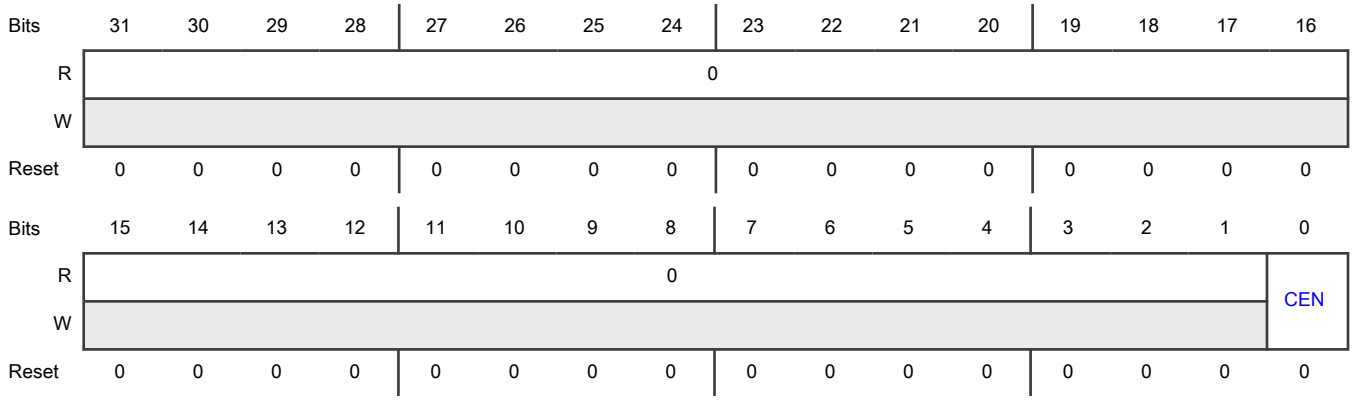
Offset

Register	Offset
CCR0	10h
CCR1	20h
CCR2	30h
CCR3	40h

Function

Enables channel n of the timer.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CEN	Channel Enable 0b - Disabled 1b - Enabled

64.7.5 Channel Interrupt (CIR0 - CIR3)

Offset

Register	Offset
CIR0	14h
CIR1	24h
CIR2	34h
CIR3	44h

Function

Indicates and clears the interrupt flag for channel n of the timer.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															CIF
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 CIF	Channel Interrupt Flag Indicates the channel IRQ is asserted due to a match on the channel. 0b - Read: IRQ is not asserted. Write: No effect. 1b - Read: IRQ is asserted. Write: Clear the flag.

64.7.6 Channel Compare (CMP0 - CMP3)

Offset

Register	Offset
CMP0	18h
CMP1	28h
CMP2	38h
CMP3	48h

Function
The compare value for channel n.

Diagram

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

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

Fields

Field	Function
31-0 CMP	Channel Compare If the channel is enabled (CCR*n*[CEN]), when the timer count (CNT) matches this value, STM asserts the channel IRQ and sets the channel interrupt flag (CIR*n*[CIF]).

64.8 Glossary

IRQ	Interrupt request

Chapter 65
Periodic Interrupt Timer (PIT)

65.1 Chip-specific PIT information

65.1.1 PIT instances and configuration

This chip has up to four instances of the PIT each with 4 PIT timers/channels, each channel being 32 bits in length.

Table 359. PIT instances

Instance	MWCT2D17S	MWCT2016S	MWCT2015S
PIT_0	Yes	Yes	Yes
PIT_1	Yes	Yes	Yes
PIT_2	Yes	No	No
PIT_3	No	No	No

Table 360. PIT instances and features

Feature	PIT_0	PIT_1	PIT_2	PIT_3
Real time interrupt (RTI)	Yes (32-bit)[1]	No	No	No
Lifetime Timer	Yes	No	No	No
Timer chaining to create a 64-bit timer	Yes	No	No	No
Number of 32-bit timer channels	4	4	4	4
Timer value unchanged by a non-destructive reset	No	No	No	No

1. The RTI on PIT_0 supports operation in STANDBY mode

The RTI shall be possible to set the timer resolution to 1us independently to other timers. The interrupt can be generated by programming LDVAL0, LDVAL1, LDVAL2 and LDVAL3 to 48 in case of 48 MHz FIRC as system clock and 40 in case of 40 MHz system clock to generate an interrupt after 1us.

65.1.2 Input Output

All the PIT instances are capable of generating periodic triggers. PIT triggers are routed to motor control IPs such as eMIOS, LCU, BCTU, ADC etc. via TRGMUX. See the TRGMUX connectivity file attached to this document for details.

65.2 Introduction

The following sections show the PIT's:

- Block diagram
- Features

65.2.1 Block diagram

The following figure shows the block diagram of PIT_RTI module.

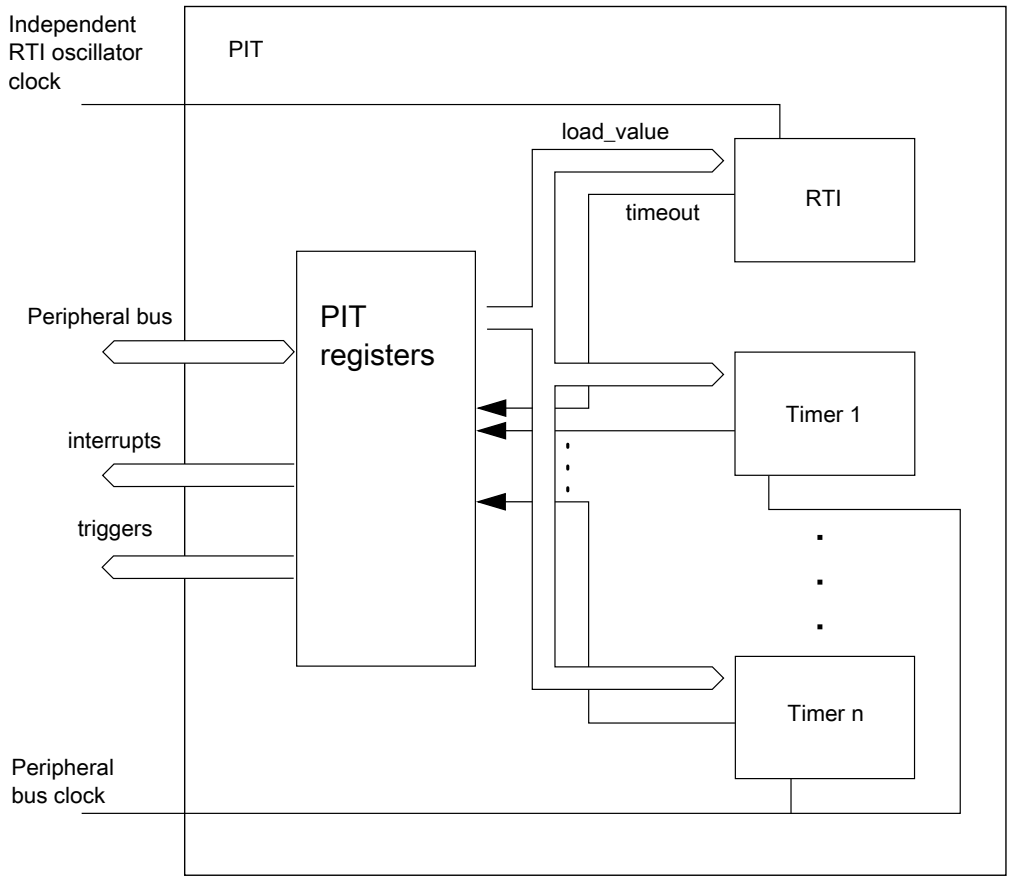


Figure 355. Block diagram of PIT_RTI

65.2.2 Features

The key features of the module are:

- One RTI timer to wakeup the CPU
- Ability of timers to generate trigger pulses
- Ability of timers to generate interrupts
- Maskable interrupts
- Option to raise RTI interrupt, even when the bus clock is switched off
- Power saving with a separate input clock for the RTI timer. All other timers share a common core clock.
- Independent timeout periods for each timer
- All the timers use a downcounter

65.3 Modes of operation

This subsection briefly describes all operating modes supported by PIT.

- Run mode

  All functional parts of the PIT are running during normal Run mode.

65.4 Functional description

This section provides the functional description of the module.

65.4.1 General operation

This section provides detailed information on the internal operation of the module. Each timer can be used to generate trigger pulses and interrupts, and each interrupt is available on a separate interrupt line. Additionally, the RTI timer can be used to wake up the processor.

65.4.1.1 Timers/RTI

The timers generate triggers at periodic intervals, when enabled. The timers load the start values as specified in their LDVAL registers, count down to 0 and then load the respective start values again. Each time a timer reaches 0, it generates a trigger pulse and sets the interrupt flag.

All interrupts can be enabled or masked by setting TCTRLn[TIE]. A new interrupt can be generated only after the previous one is cleared. Because in the case of the RTI, clearing the interrupt crosses clock domains, a minimum load value of 32 must be maintained.

If desired, the current counter value of the timer can be read via the CVAL registers. The value of the RTI counter can be delayed considerably, as it is synchronized to the bus clock from the RTI clock domain.

The counter period can be restarted by first disabling and then enabling the timer with TCTRLn[TEN]. See the following figure.

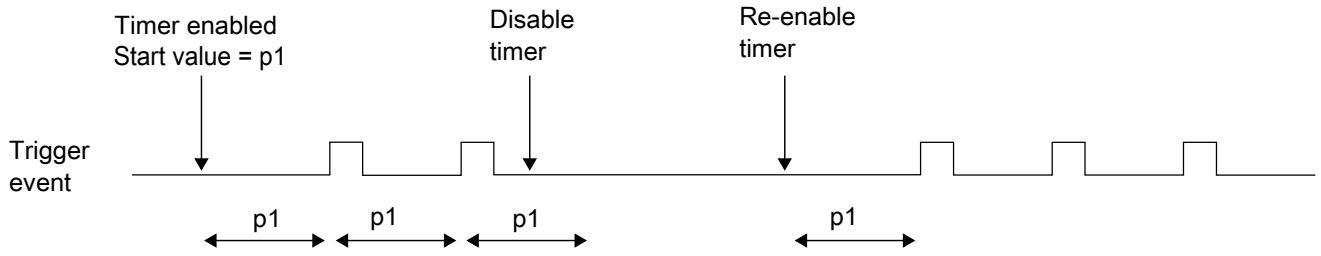


Figure 356. Stopping and starting a timer

The counter period of a running timer can be modified by first disabling the timer, setting a new load value, and then enabling the timer again. See the following figure.

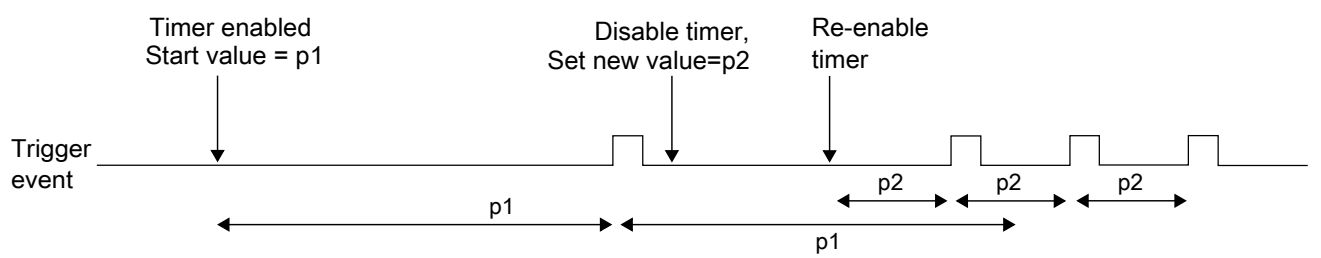


Figure 357. Modifying running timer period

It is also possible to change the counter period without restarting the timer, by writing LDVAL with the new load value. This value then loads after the next trigger event. See the following figure.

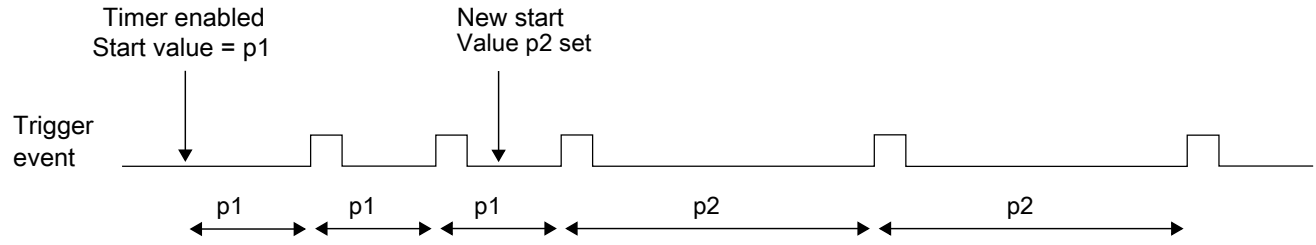


Figure 358. Dynamically setting a new load value

NOTE

- If MCR[FRZ] is written as 1 and when the timer is nearing 0 (CVALn =0x0), the debug mode command may not make it to the IP before the timer expires and generates a trigger.
- Debug Mode will be ignored if CVALn =0x0, but the trigger will remain asserted until the mode is removed. The user is recommended to exit Debug mode and then clear the interrupt TFLGn[TIF].
- If the timers are to be freezed for debug, sufficient time must be ensured for the IP to react.

65.4.1.2 Debug mode

In the Debug mode, the timers are frozen based on MCR[FRZ]. This is intended to aid software development, allowing the developer to halt the processor, investigate the current state of the system, for example, the timer values, and then continue the operation.

65.4.2 Interrupts

All the timers support interrupt generation. The RTI is typically used for system wakeup, but can be used for interrupt generation as well.

Timer interrupts can be enabled by setting TCTRLn[TIE].

TFLGn[TIF] are set to 1 when a timeout occurs on the associated timer, and are cleared to 0 by writing a 1 to the corresponding TFLGn[TIF].

The PIT_RTI generates an RTI when the selected interrupt time period elapses. The RTI is disabled locally by setting TCTCRLn[TIE] to zero. The real-time interrupt flag (TIF) is set to 1 when a timeout occurs, and is cleared by writing a 1 to the corresponding TFLGn[TIF]. The flag is set regardless of whether the interrupt is enabled.

The RTI can be used for periodic wakeup from a STANDBY mode. It can also be used to generate a general-purpose interrupt.

65.4.3 Chained timers

When a timer has chain mode enabled, it counts after the previous timer has expired. So if timer n-1 counts down to 0, counter n decrements the value by one. This allows to chain some of the timers together to form a longer timer. The first timer (timer 0) cannot be chained to any other timer. The same applies to RTI. See Example configuration for chained timers for details.

65.5 Initialization and application information

In the example configuration:

- The PIT clock has a frequency of 50 MHz.
- The RTI clock has a frequency of 10 MHz.
- The RTI creates a wakeup interrupt every 500 ms.
- Timer 1 creates an interrupt every 5.12 ms.
- Timer 3 creates a trigger event every 30 ms.

The PIT module must be activated by writing a 0 to MCR[MDIS].

The 50 MHz clock frequency equates to a clock period of 20 ns and the 10 MHz frequency equates to a clock period of 100 ns. Therefore, the RTI timer needs to trigger every 500 ms/100 ns = 5,000,000 cycles. Timer 1 needs to trigger every 5.12 ms/20 ns = 256,000 cycles and Timer 3 every 30 ms/20 ns = 1,500,000 cycles. The value for the LDVAL register trigger is calculated as:

LDVAL trigger = (period / clock period) -1

This means RTI LDVAL is written with 004C4B3F hex; LDVAL1 and LDVAL3 must be written with 0x0003E7FF and 0x0016E35F, respectively.

To generate the wakeup interrupt, the interrupt line must be enabled by writing 1 to the RTI TIE bit in the TCTRL register. To start the RTI, the TEN in the RTI TCTRL register must also be set.

The interrupt for Timer 1 is enabled by setting TCTRL1[TIE]. The timer is started by writing 1 to TCTRL1[TEN].

Timer 3 shall be used only for triggering. Therefore, Timer 3 is started by writing a 1 to TCTRL3[TEN]. But, TCTRL3[TIE] stays at 0.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;
```

```
// RTI
PIT_RTI_LDVAL = 0x004C4B3F; // setup RTI for 5000000 cycles
PIT_RTI_TCTRL = PIT_TIE; // let RTI generate interrupts
PIT_RTI_TCTRL |= PIT_TEN; // start RTI
```

```
// Timer 1
PIT_LDVAL1 = 0x0003E7FF; // setup timer 1 for 256000 cycles
PIT_TCTRL1 = TIE; // enable Timer 1 interrupts
PIT_TCTRL1 |= TEN; // start Timer 1
```

```
// Timer 3
PIT_LDVAL3 = 0x0016E35F; // setup timer 3 for 1500000 cycles
PIT_TCTRL3 |= TEN; // start Timer 3
```

65.6 Example configuration for chained timers

In the example configuration:

- The PIT clock has a frequency of 100 MHz.
- Timers 1 and 2 are available.
- An interrupt is raised every 1 minute.

The PIT module needs to be activated by writing a 0 to MCR[MDIS].

The 100 MHz clock frequency equates to a clock period of 10 ns, so the PIT needs to count for 6000 million cycles, which is more than a single timer can do. So, Timer 1 is set up to trigger every 6 s (600 million cycles). Timer 2 is chained to Timer 1 and programmed to trigger 10 times.

The value for the LDVAL register trigger is calculated as number of cycles-1, so LDVAL1 receives the value 0x23C345FF and LDVAL2 receives the value 0x00000009.

The interrupt for Timer 2 is enabled by setting TCTRL2[TIE], the Chain mode is activated by setting TCTRL2[CHN], and the timer is started by writing a 1 to TCTRL2[TEN].

TCTRL1[TEN] needs to be set, and TCTRL1[CHN] and TCTRL1[TIE] are cleared.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;
```

```
// Timer 2
PIT_LDVAL2 = 0x00000009; // setup Timer 2 for 10 counts
PIT_TCTRL2 = TIE; // enable Timer 2 interrupt
PIT_TCTRL2 |= CHN; // chain Timer 2 to Timer 1
PIT_TCTRL2 |= TEN; // start Timer 2
```

```
// Timer 1
PIT_LDVAL1 = 0x23C345FF; // setup Timer 1 for 600 000 000 cycles
PIT_TCTRL1 = TEN; // start Timer 1
```

65.7 Example configuration for the Lifetime Timer

To configure the Lifetime Timer, channels 0 and 1 need to be chained together.

First, the PIT module needs to be activated by writing a 0 to the MDIS bit in the CTRL register, and then the LDVAL registers need to be set to the maximum value.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;
```

```
// Timer 1
PIT_LDVAL1 = 0xFFFFFFFF; // setup timer 1 for maximum counting period
PIT_TCTRL1 = 0x0; // disable timer 1 interrupts
PIT_TCTRL1 |= CHN; // chain timer 1 to timer 0
PIT_TCTRL1 |= TEN; // start timer 1
```

```
// Timer 0
PIT_LDVAL0 = 0xFFFFFFFF; // setup timer 0 for maximum counting period
PIT_TCTRL0 = TEN; // start timer 0
```

To access the Lifetime Timer, read first LTMR64H and then LTMR64L.

```
current_uptime = PIT_LTMR64H<<32;
current_uptime = current_uptime + PIT_LTMR64L;
```

65.8 PIT register descriptions

This section provides a detailed description of all registers accessible in the PIT_RTI module.

- See the chip-specific PIT information for the number of PIT channels used in this MCU.
- The RTI registers should be programmed only when the RTI clock is running.

NOTE

When a chip has more than one instance of the PIT, the different instances might have different feature and register sets. Please see the configuration information.

65.8.1 PIT memory map

PIT_0 base address: 400B_0000h

PIT_1 base address: 400B_4000h

PIT_2 base address: 402F_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PIT Module Control Register (MCR)	32	RW	See section
E0h	PIT Upper Lifetime Timer Register (LTMR64H)	32	R	0000_0000h
E4h	PIT Lower Lifetime Timer Register (LTMR64L)	32	R	0000_0000h
ECh	RTI Timer Load Value Sync Status Register (RTI_LDVAL_STAT)	32	RW	0000_0000h
F0h	Timer Load Value Register (RTI_LDVAL)	32	RW	0000_0000h
F4h	Current Timer Value Register (RTI_CVAL)	32	R	0000_0000h
F8h	Timer Control Register (RTI_TCTRL)	32	RW	0000_0000h
FCh	Timer Flag Register (RTI_TFLG)	32	RW	0000_0000h
100h	Timer Load Value Register (LDVAL0)	32	RW	0000_0000h
104h	Current Timer Value Register (CVAL0)	32	R	0000_0000h
108h	Timer Control Register (TCTRL0)	32	RW	0000_0000h
10Ch	Timer Flag Register (TFLG0)	32	RW	0000_0000h
110h	Timer Load Value Register (LDVAL1)	32	RW	0000_0000h
114h	Current Timer Value Register (CVAL1)	32	R	0000_0000h
118h	Timer Control Register (TCTRL1)	32	RW	0000_0000h
11Ch	Timer Flag Register (TFLG1)	32	RW	0000_0000h
120h	Timer Load Value Register (LDVAL2)	32	RW	0000_0000h
124h	Current Timer Value Register (CVAL2)	32	R	0000_0000h
128h	Timer Control Register (TCTRL2)	32	RW	0000_0000h
12Ch	Timer Flag Register (TFLG2)	32	RW	0000_0000h
130h	Timer Load Value Register (LDVAL3)	32	RW	0000_0000h
134h	Current Timer Value Register (CVAL3)	32	R	0000_0000h
138h	Timer Control Register (TCTRL3)	32	RW	0000_0000h
13Ch	Timer Flag Register (TFLG3)	32	RW	0000_0000h

65.8.2 PIT Module Control Register (MCR)

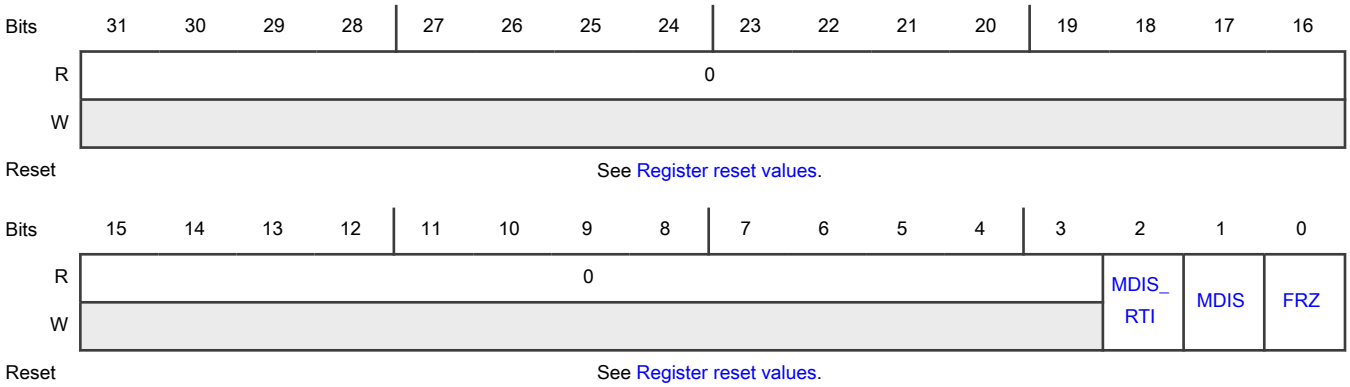
Offset

Register	Offset
MCR	0h

Function

This register enables or disables the PIT timer clocks and controls the timers when the PIT enters the Debug mode.

Diagram



Register reset values

Register	Reset value
MCR	PIT_0: 0000_0006h PIT_1,PIT_2: 0000_0002h

Fields

Field	Function
31-3 —	Reserved
2 MDIS_RTI	Module Disable for RTI Disables the RTI timer. You must write 1 to this field before setting up an RTI. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PIT_0</td><td>MCR</td><td>—</td></tr><tr><td>PIT_1</td><td>—</td><td>MCR</td></tr><tr><td>PIT_2</td><td>—</td><td>MCR</td></tr></table> 0b - Clock for RTI timers is enabled 1b - Clock for RTI timers is disabled
1 MDIS	Module Disable for PIT Disables the standard timers. The RTI timer is not affected by this field. You must write 1 to this field before setting up anything. 0b - Clock for standard PIT timers is enabled. 1b - Clock for standard PIT timers is disabled.
0 FRZ	Freeze Allows the timers to be stopped when the device enters the Debug mode. 0b - Timers continue to run in Debug mode. 1b - Timers are stopped in Debug mode.

65.8.3 PIT Upper Lifetime Timer Register (LTMR64H)

Offset

Register	Offset
LTMR64H	E0h

Function

This register is intended for applications that chain timer 0 and timer 1 to build a 64-bit Lifetime Timer.

NOTE

Each module instance supports a different number of registers.

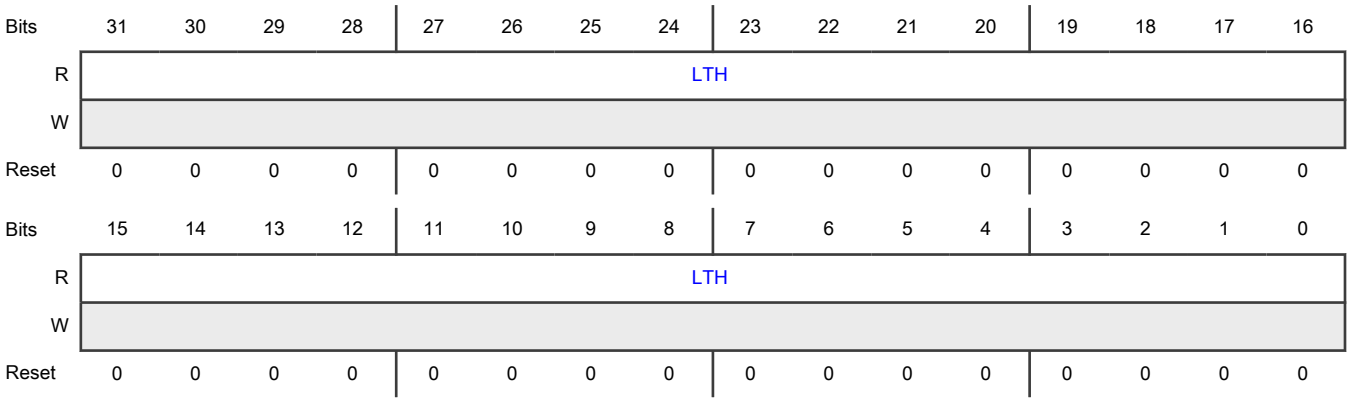
Instance	Register supported	Register not supported
PIT_0	LTMR64H	—
PIT_1	—	LTMR64H

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
PIT_2	—	LTMR64H

Diagram



Fields

Field	Function
31-0 LTH	Life Timer value Shows the timer value of timer 1. If this register is read at a time t1, LTMR64L shows the value of timer 0 at time t1.

65.8.4 PIT Lower Lifetime Timer Register (LTMR64L)

Offset

Register	Offset
LTMR64L	E4h

Function

This register is intended for applications that chain timer 0 and timer 1 to build a 64-bit Lifetime Timer.

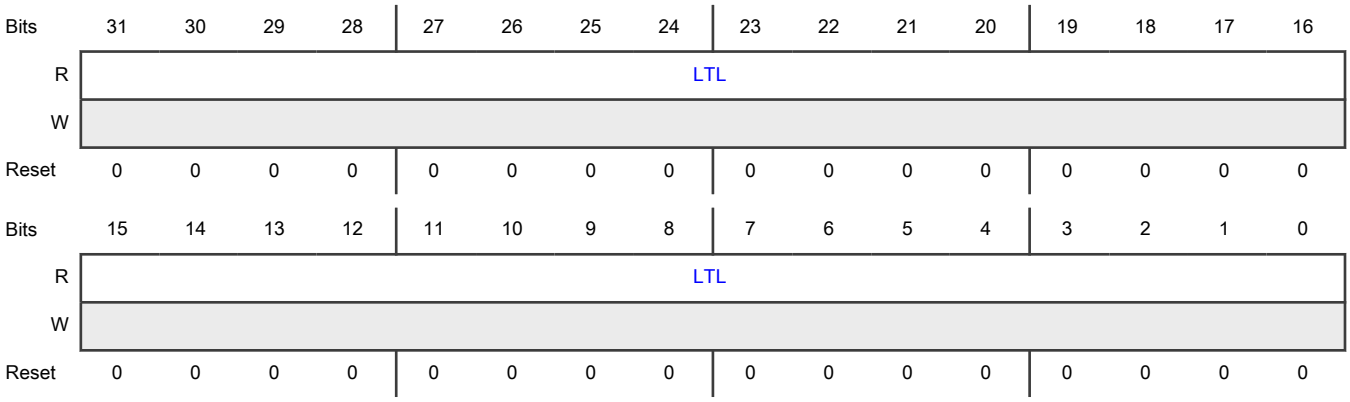
To use LTMR64H and LTMR64L, timer 0 and timer 1 need to be chained. To obtain the correct value, first read LTMR64H and then LTMR64L. The value for the LTMR64H register is set to CVAL1 register at the time of the first access and the value of the LTMR64L register is set to CVAL0 register at first access. Therefore, the application is not affected by the carry-over effects of the running counter.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PIT_0	LTMR64L	—
PIT_1	—	LTMR64L
PIT_2	—	LTMR64L

Diagram



Fields

Field	Function
31-0 LTL	Life Timer value Shows the value of timer 0 at the time LTMR64H was last read. It will only update if LTMR64H is read.

65.8.5 RTI Timer Load Value Sync Status Register (RTI_LDVAL_STAT)

Offset

Register	Offset
RTI_LDVAL_STAT	ECh

Function

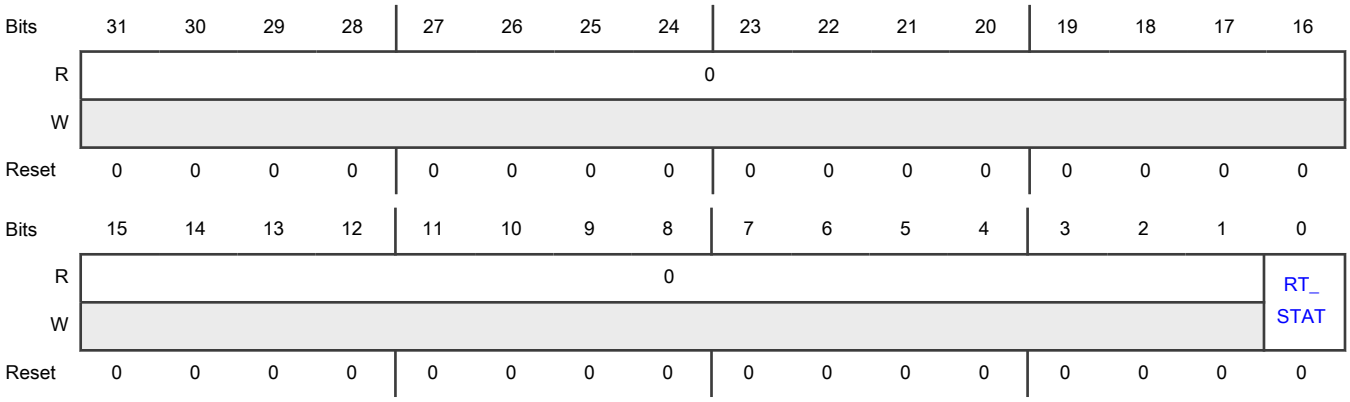
These registers gives the RTI timer load synchronization status. In the case of the RTI timer load, it will take several cycles until this value is synchronized into the RTI clock domain. This register gives the status of the new value loaded in the RTI timer load register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PIT_0	RTI_LDVAL_STAT	—
PIT_1	—	RTI_LDVAL_STAT
PIT_2	—	RTI_LDVAL_STAT

Diagram



Fields

Field	Function
31-1 —	Reserved
0 RT_STAT	RTI Timer Load Value Sync Status RTI timer load value loaded in RTI_LDVAL register. **NOTE** Any attempt to write will clear this field. 0b - RTI_LDVAL register not loaded. 1b - RTI_LDVAL register updated with new value.

65.8.6 Timer Load Value Register (RTI_LDVAL)

Offset

Register	Offset
RTI_LDVAL	F0h

Function

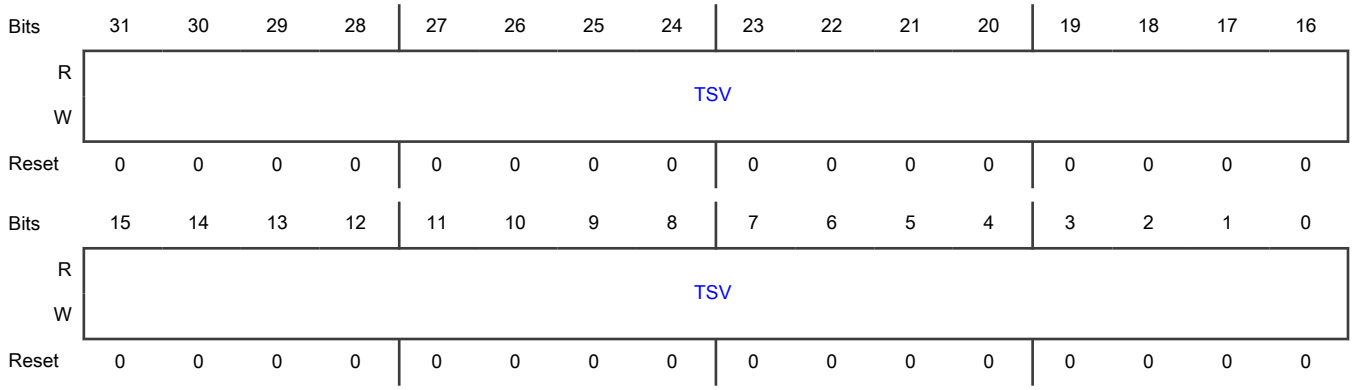
These registers select the timeout period for the timer interrupts. In the case of the RTI, it will take several cycles until this value is synchronized into the RTI clock domain. For all other timers, the value change is visible immediately. The synchronization mechanism allows 0 wait states in this case.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PIT_0	RTI_LDVAL	—
PIT_1	—	RTI_LDVAL
PIT_2	—	RTI_LDVAL

Diagram



Fields

Field	Function
31-0 TSV	Timer Start Value Sets the timer start value. The timer counts till it reaches 0; then it generates an interrupt and loads the register value again. Writing a new value to this register does not restart the timer; instead the value loads after the timer expires. To abort the current cycle and start a timer period with a new value, the timer must be disabled and enabled again. **NOTE** The RTI timer must not be set to a value lower than 32 cycles, otherwise interrupts may be lost, as it takes several cycles to clear the RTI interrupt. For the other timers, this limit does not apply; however, there are practical limits because the processor requires several cycles to service an interrupt.

65.8.7 Current Timer Value Register (RTI_CVAL)

Offset

Register	Offset
RTI_CVAL	F4h

Function

These registers indicate the current timer position. For the RTI timers, they show a value that is several cycles old because it originates from a potentially different clock domain.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PIT_0	RTI_CVAL	—
PIT_1	—	RTI_CVAL
PIT_2	—	RTI_CVAL

Diagram

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

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

Fields

Field	Function
31-0 TVL	Current Timer Value Represents the current timer value, if the timer is enabled. **NOTE** • If the timer is disabled, do not use this field because its value is unreliable. • The timer uses a downcounter. The timer values are frozen in the Debug mode if MCR[FRZ] is set.

65.8.8 Timer Control Register (RTI_TCTRL)

Offset

Register	Offset
RTI_TCTRL	F8h

Function

These registers contain the control bits for each timer. The RTI might take several RTI clock cycles to get enabled or updated. Hence, you must wait for at least three RTI clock cycles after RTI configuration.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PIT_0	RTI_TCTRL	—
PIT_1	—	RTI_TCTRL
PIT_2	—	RTI_TCTRL

Diagram

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

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

Fields

Field	Function
31-3 —	Reserved
2 CHN	Chain Mode The RTI timer cannot be chained; writing to this bit has no impact.
1	Timer Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TIE	When an interrupt is pending or if RTI_TFLG[TIF] is set, enabling the interrupt immediately causes an interrupt event. To avoid this, the associated RTI_TFLG[TIF] flag must be cleared first. 0b - Interrupt requests from the RTI timers are disabled. 1b - Interrupt is requested whenever TIF is set.
0 TEN	Timer Enable Bit This bit enables or disables the timer. 0b - RTI timer is disabled. 1b - RTI timer is enabled.

65.8.9 Timer Flag Register (RTI_TFLG)

Offset

Register	Offset
RTI_TFLG	FCh

Function

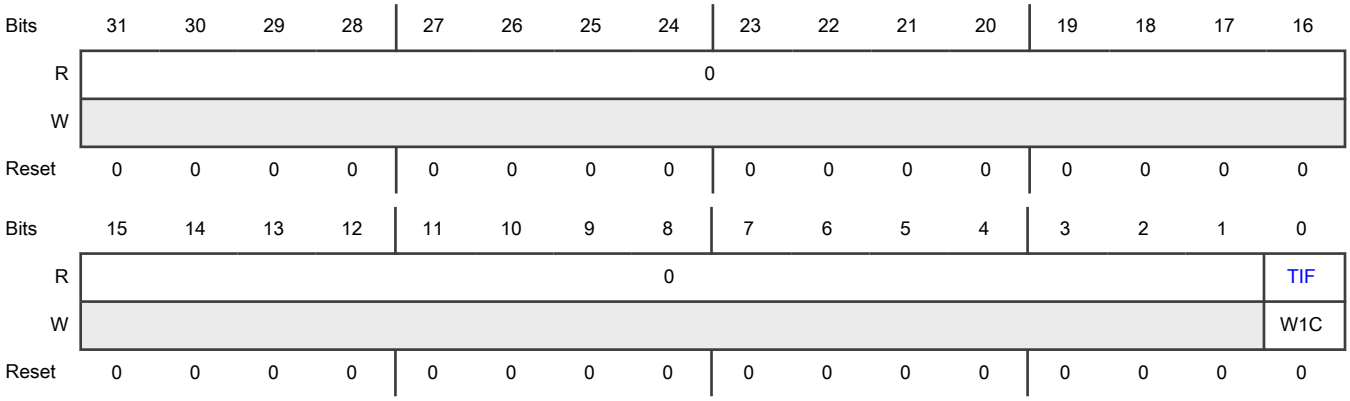
These registers hold the PIT interrupt flags.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PIT_0	RTI_TFLG	—
PIT_1	—	RTI_TFLG
PIT_2	—	RTI_TFLG

Diagram



Fields

Field	Function
31-1 —	Reserved
0 TIF	Timer Interrupt Flag TIF is set to 1 at the end of the timer period. This flag can be cleared only by writing it with 1 . Writing 0 has no effect. If enabled (TIE = 1), TIF causes an interrupt request. 0b - Time-out has not yet occurred. 1b - Time-out has occurred.

65.8.10 Timer Load Value Register (LDVAL0 - LDVAL3)

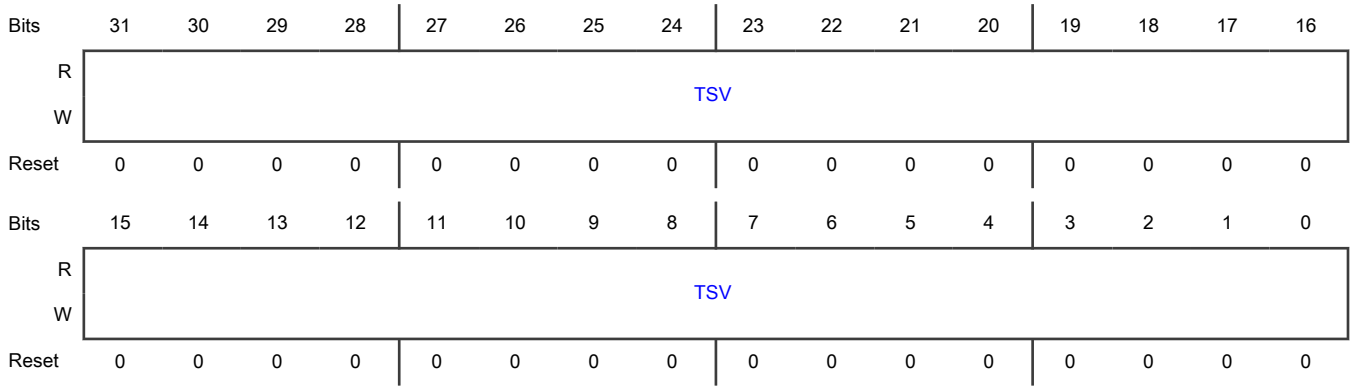
Offset

Register	Offset
LDVAL0	100h
LDVAL1	110h
LDVAL2	120h
LDVAL3	130h

Function

These registers select the timeout period for the timer interrupts.

Diagram



Fields

Field	Function
31-0 TSV	Timer Start Value Sets the timer start value. The timer counts down until it reaches 0, then generates an interrupt and loads this register value again. Writing a new value to this register does not restart the timer; instead the value is loaded after the timer expires. To abort the current cycle and start a timer period with the new value, the timer must be disabled and enabled again.

65.8.11 Current Timer Value Register (CVAL0 - CVAL3)

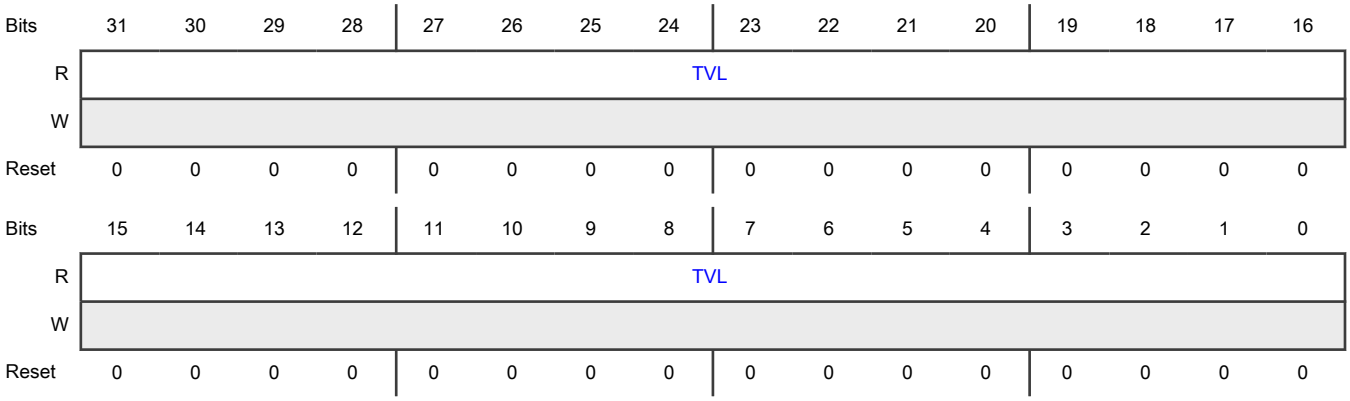
Offset

Register	Offset
CVAL0	104h
CVAL1	114h
CVAL2	124h
CVAL3	134h

Function

These registers indicate the current timer position.

Diagram



Fields

Field	Function
31-0 TVL	Current Timer Value Represents the current timer value, if the timer is enabled. **NOTE** • If the timer is disabled, do not use this field because its value is unreliable. • The timer uses a downcounter. The timer values are frozen in the Debug mode if MCR[FRZ] is set.

65.8.12 Timer Control Register (TCTRL0 - TCTRL3)

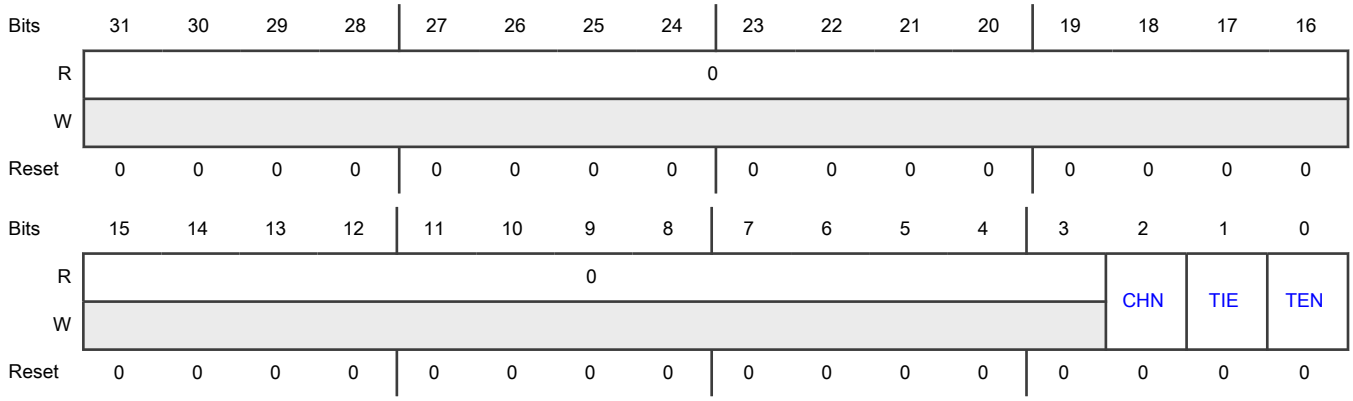
Offset

Register	Offset
TCTRL0	108h
TCTRL1	118h
TCTRL2	128h
TCTRL3	138h

Function

These registers contain the control bits for each timer.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 CHN	Chain Mode When activated, timer n-1 needs to expire before timer n (n is > 0) can decrement by 1. Timer 0 cannot be chained. The RTI timer cannot be chained. 0b - Timer is not chained. 1b - Timer is chained to a previous timer. For example, for channel 2, if this field is set, Timer 2 is chained to Timer 1.
1 TIE	Timer Interrupt Enable When an interrupt is pending, or if TFLGn[TIF] is set, enabling the interrupt causes an interrupt event. To avoid this, the associated TFLGn[TIF] must be cleared first. 0b - Interrupt requests from Timer n are disabled. 1b - Interrupt is requested whenever TIF is set.
0 TEN	Timer Enable Enables or disables the timer. 0b - Timer n is disabled. 1b - Timer n is enabled.

65.8.13 Timer Flag Register (TFLG0 - TFLG3)

Offset

Register	Offset
TFLG0	10Ch

Table continues on the next page...

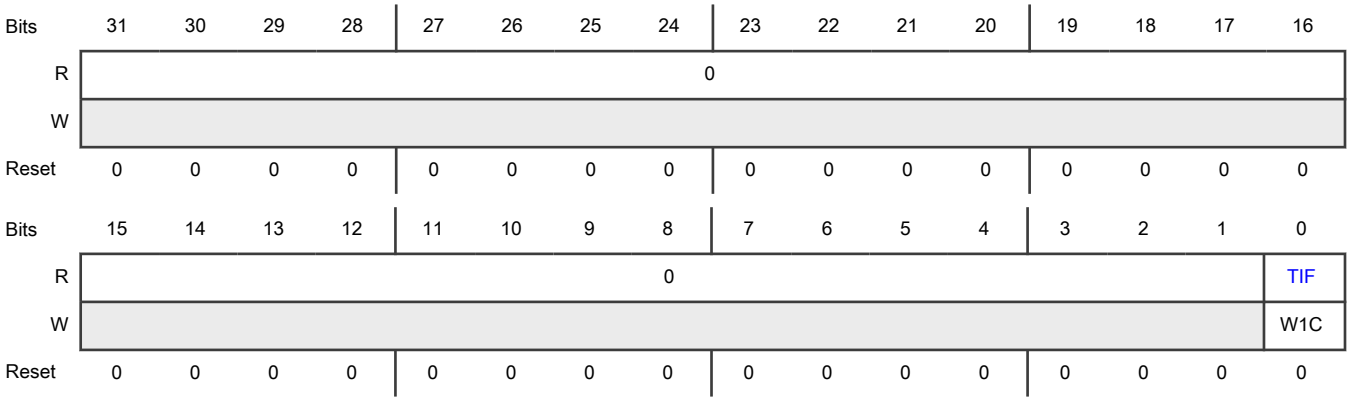
Table continued from the previous page...

Register	Offset
TFLG1	11Ch
TFLG2	12Ch
TFLG3	13Ch

Function

These registers hold the PIT interrupt flags.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 TIF	Timer Interrupt Flag Sets to 1 at the end of the timer period. Writing 1 to this flag clears it and writing 0 has no effect. If enabled, or, when TCTRLn[TIE] = 1, TIF causes an interrupt request. 0b - Timeout has not yet occurred. 1b - Timeout has occurred.

65.9 Glossary

RTI Real-time interrupt

Chapter 66
Real Time Clock (RTC)

66.1 Chip-specific RTC information

66.1.1 RTC instances and configuration

The chip contains one instance of RTC (Real Time Clock) timer and API (Autonomous Periodic Interrupt) timer, where both can perform 32-bit comparisons.

The RTC is present in always ON domain, hence available in RUN mode as well as in STANDBY mode. Both RTC and API timers can generate interrupts as well as wakeup from low power modes.

RTC supports a resolution of 1µs with high speed internal oscillator clock (1/48MHz (FIRC) * 48 (RTC counter) = 1µs)

This chip supports seamless RTC operation across functional reset with the clock sources SIRC and SXOSC.

NOTE

SXOSC is not present in MWCT2015S, thus RTC will run from SIRC clock.

API has the capability to change the Timer compare value (APIVAL) independently without stopping the timer.

API can also be used in conjunction with the Comparator module. In STANDBY mode configuration, the API is configured to generate a START / NEXT type signal to inform the Comparator module that a 'compare' must be carried out.

66.3 Introduction

The Real-Time Clock (RTC) is a free-running counter used for time keeping applications. The RTC can be configured to generate an interrupt at a pre-defined interval independent of the mode of operation (run mode or low power mode). If in a low power mode, the RTC interval is reached, the RTC first generates a wakeup and then asserts the interrupt request. The RTC also supports an API function used to generate a periodic wakeup request to exit a low-power mode or an interrupt request.

66.3.1 Features

RTC features include:

- 32-bit counter
- Selectable counter clock sources (IRCs and OSCs)
  - Clock source 0
  - Clock source 1
  - Clock source 2
  - Clock source 3
- Optional 512 prescaler and optional 32 prescaler to run the 32 bit counter.
- RTC interrupt with interrupt enable.
- Counter runs in all modes of operation.
- RTC counter is reset when the counter is disabled by software and by reset to RTC block.
- Autonomous periodic interrupt support includes:
  - 32-bit compare value to support range of wakeup intervals/interrupts
  - API logic has a separate enable to support changing compare value while RTC is running
  - API interrupt with interrupt enable
  - Operates in all modes

— API compare value can be modified while RTC is running

- Optional interrupt for RTC match, API match, and RTC rollover.

66.3.2 Block diagram

The following figure shows Clock gating for RTC clocks.

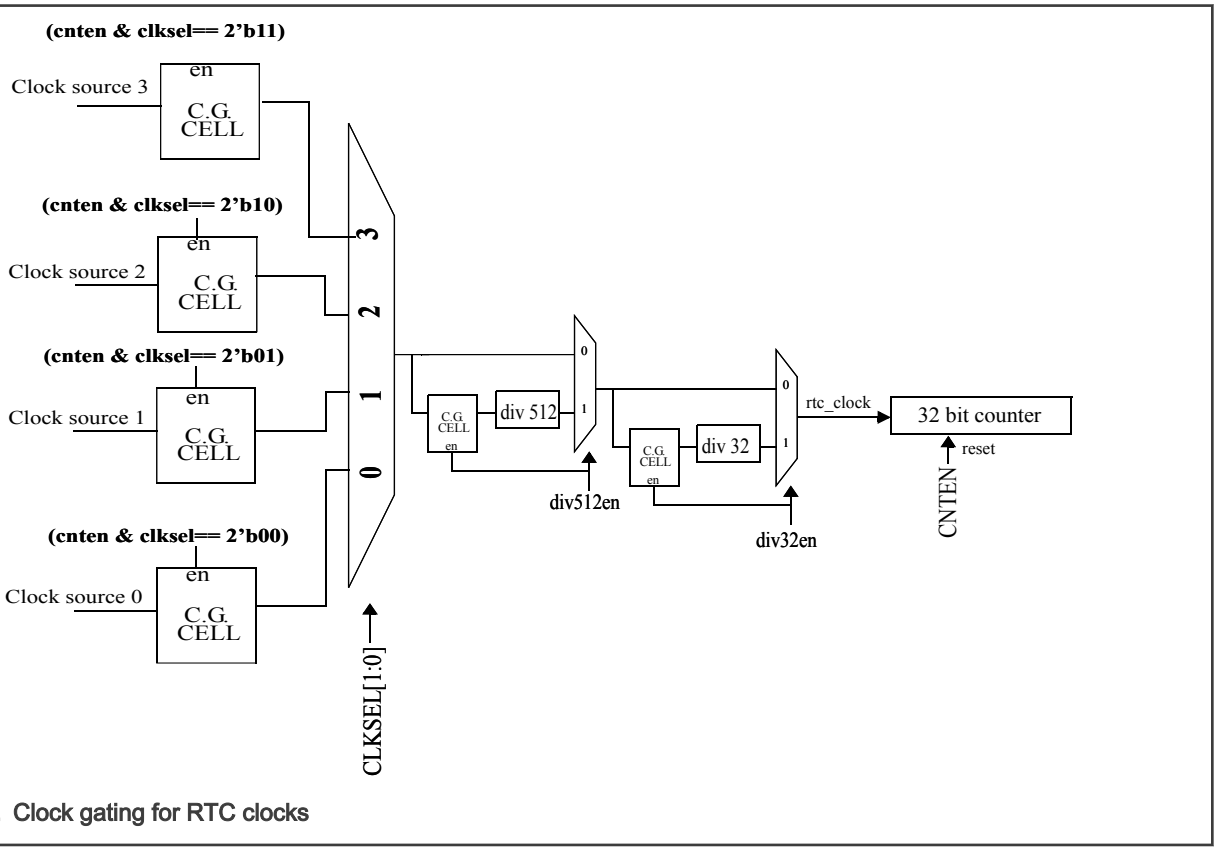


Figure 359. Clock gating for RTC clocks

The following figure shows the block diagram of RTC.

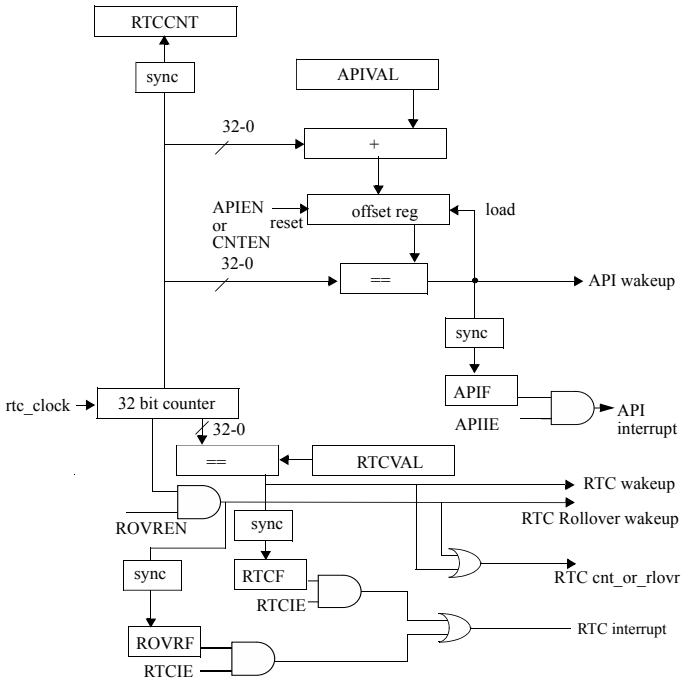


Figure 360. Block diagram

66.3.3 Modes of operation

66.3.3.1 Functional mode

There are two functional modes of operation for RTC: normal operation and low-power mode. In normal operation, all RTC registers can be read or written . The RTC/API and associated interrupts are optionally enabled. In low-power mode, the bus interface is disabled. The RTC/API is enabled (if enabled prior to entry into low-power mode).

66.3.3.2 Debug mode

On entering into the debug mode, the RTC counter freezes on the last valid count if the FRZEN is set. On exit from debug mode, counter continues from the frozen value.

66.4 Functional description

66.4.1 RTC

The RTC consists of a 32-bit free running counter enabled with the CNTEN bit (CNTEN, when negated, asynchronously resets the counter and synchronously enables the counter when enabled). After disabling CNTEN, RTCVAL, APIVAL, needs to be written again for desired functionality. The value of the counter may be read via the RTC Counter register (RTCCNT) register. Note that because of clock synchronization, the RTC Counter register (RTCCNT) value may represent a previous counter value. The difference between the counter and the read value depends on the ratio of counter clock and bus clock. Maximum possible difference between the two is 6 count values.

The clock source to the counter is selected with the CLKSEL field, which gives four options for clocking the RTC/API. The four clock sources are assumed to be on these:Clock source 0, Clock source 1, Clock source 2, and Clock source 3. The output of the clock mux can be optionally divided by a combination of 512 and 32 to give various count periods for different clock sources. Note that the CNTEN bit should be disabled when the RTC/API clock source is switched.

When the RTC Counter register (RTCCNT) counter value for counter bits 31-0 match the 32-bit value in the RTCVAL field, then the RTCF interrupt flag bit is set (after proper clock synchronization). If the RTCIE interrupt enable bit is set, then the RTC interrupt request is generated. RTC Compare value register (RTCVAL) register can be written only when INV_RTC bit is clear. Initially INV_RTC=0, and hence RTC Compare value register (RTCVAL) can be written once and hence INV_RTC gets set. This bit can now be cleared only by enabling the RTC counter. After the counter is enabled, RTC Compare value register (RTCVAL) can be written anytime, until RTC is disabled again. RTC Compare value register (RTCVAL) is first synchronized to the RTC clock domain, therefore, if RTC Compare value register (RTCVAL) is updated at the point where a counter match is due to happen in the next 2-3 RTC clocks because of previous RTC Compare value register (RTCVAL), the RTCF flag is set. However, if the RTC Compare value register (RTCVAL) is updated at the point where no counter match is due as per the previous RTC Compare value register (RTCVAL), the RTCF flag is set when the counter matches the new RTC Compare value register (RTCVAL). If there is a match when in the low-power mode, then the RTC first generates a wakeup request to force a wakeup to run mode,and then the RTCF flag is set.

If RTC Compare value register (RTCVAL) is updated just after counter match, new RTC Compare value register (RTCVAL) value should not be within next 6 RTC counter values.

A rollover wakeup and/or interrupt can be generated when the RTC transitions from a count of 0xFFFF_FFFF to 0x0000_0000. The rollover flag is enabled by setting the ROVREN bit. If RTCIE is enabled, an interrupt request is generated for an RTC counter rollover. If system is in low power mode, RTC counter rollover with this bit causes a wakeup from the low-power mode.

Both rollover wakeup and RTC wakeup gets asynchronously de-asserted with disabling of CNTEN.

All the flags and counter values are synchronized with the Bus clock. It is assumed that the Bus clock and RTC clock selected through CLKSEL follows the below relation:

Bus clock >= (1.5 * RTC clock)/(div_factor)

- if both DIV32EN and DIV512EN bits are disabled, div_factor = 1
- if DIV32EN=1 and DIV512EN=0, div_factor = 32
- if DIV32EN=0 and DIV512EN=1, div_factor = 512
- if both DIV32EN and DIV512EN bits are enabled, div_factor = 512*32 = 16384

In case, RTC wakeup's are used as a wakeup source, Bus clock should be disabled after enabling the required wakeup and ensuring sufficient time gap (few Bus clock or RTC clock cycles, whichever is slower) between Bus clock disabling and wakeup event.

66.4.2 API functional description

Setting APIEN bit enables the autonomous interrupt function. The 32-bit APIVAL field selects the time interval for triggering an interrupt and/or wakeup event. Because the RTC is a free-running counter, the APIVAL is added to the current count to calculate an offset. When the counter reaches the offset count, an interrupt and/or wakeup request is generated. Then the offset value is recalculated and again re-triggers a new request when the new value is reached. API function is enabled only when CNTEN and APIEN bits are asserted and APIVAL is non-zero. Also, APIVAL can be updated anytime. After APIVAL is updated, the first API interrupt is generated according to the previous value. From the second interrupt onwards, the API interrupt is generated with the new APIVAL. When a compare is reached, the APIF interrupt flag bit is set (after proper clock synchronization). If the APIIE interrupt enable bit is set, then the API interrupt request is generated. If there is a match while being in the low-power mode, then the API first generates a wakeup request to force a wakeup into normal operation, and then the APIF flag is set.

When the CNTEN is de-asserted, the API function is reset , though wakeup API is not asynchronously de-asserted with CNTEN. If APIEN is disabled when counter matches the offset, APIF can be missed.

66.5 Recommended programming flow

1. Program RTCVAL register with the value greater than 4 if RTCF related functionality is required.
2. Program APIVAL register with the value greater than 4 if API functionality is required.
3. Program all fields (as required) of RTC Control register (RTCC) register with CNTEN=1. After setting CNTEN field, counter starts running and RTC/API functionality will be active as per the configurations.

4. If RTCVAL or APIVAL needs to be updated during run, check corresponding INV_RTC or INV_API bit is cleared before writing and do meet all the restrictions of the corresponding register.
5. If CLKSEL or DIV32EN or DIV512EN needs to be updated, first update RTC Control register (RTCC) register with CNTEN=0 (when all INV_RTC or INV_API are 0) keeping other field values same. Wait for minimum 3 RTC clock cycles so that CNTEN will be synchronized to RTC clock domain. Then write the RTC Control register (RTCC) register with new configuration required with CNTEN=1.

66.6 RTC register descriptions

The RTC registers are listed in this section.

NOTE

Address offset - 0x18h should not be accessed by application as corresponding feature/s are not available. Therefore, transfer error will not be generated at this offset.

NOTE

XFR error will be generated when RTCSUPV is accessed in user mode, any other register is accessed in user mode when SUPV bit is set, write attempt is made for RTCCNT register and any register accessed out of address range.

66.6.1 RTC memory map

RTC base address: 4028_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	RTC Supervisor control register (RTCSUPV)	32	RW	8000_0000h
4h	RTC Control register (RTCC)	32	RW	0000_0000h
8h	RTC Status register (RTCS)	32	RW	0000_0000h
Ch	RTC Counter register (RTCCNT)	32	R	0000_0000h
10h	API Compare value register (APIVAL)	32	RW	0000_0000h
14h	RTC Compare value register (RTCVAL)	32	RW	0000_0000h

66.6.2 RTC Supervisor control register (RTCSUPV)

Offset

Register	Offset
RTCSUPV	0h

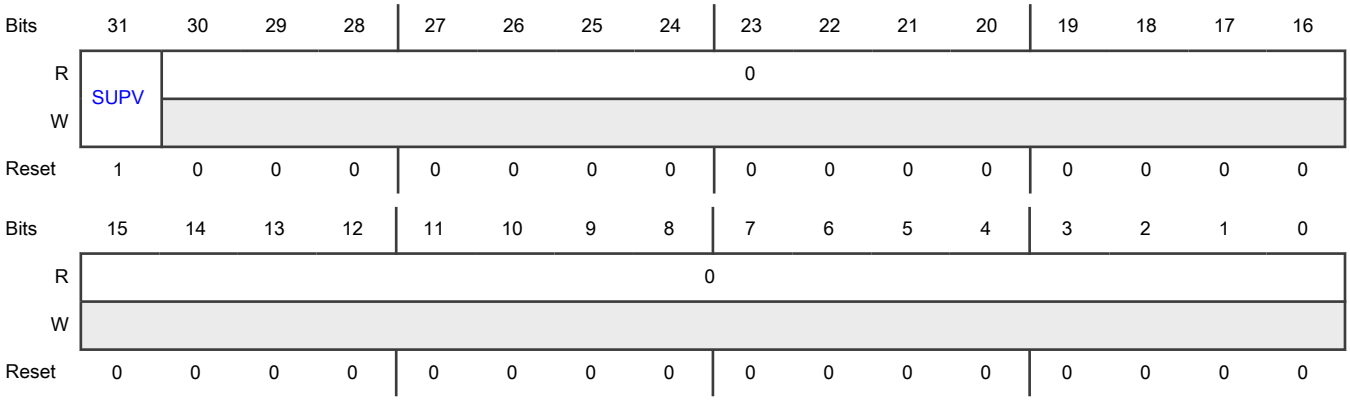
Function

The RTCSUPV register contains the SUPV bit that determines whether other registers are accessible in supervisor mode or user mode.

NOTE

You can access this register only in Supervisor mode, and you must write a value only to the SUPV field of the register.

Diagram



Fields

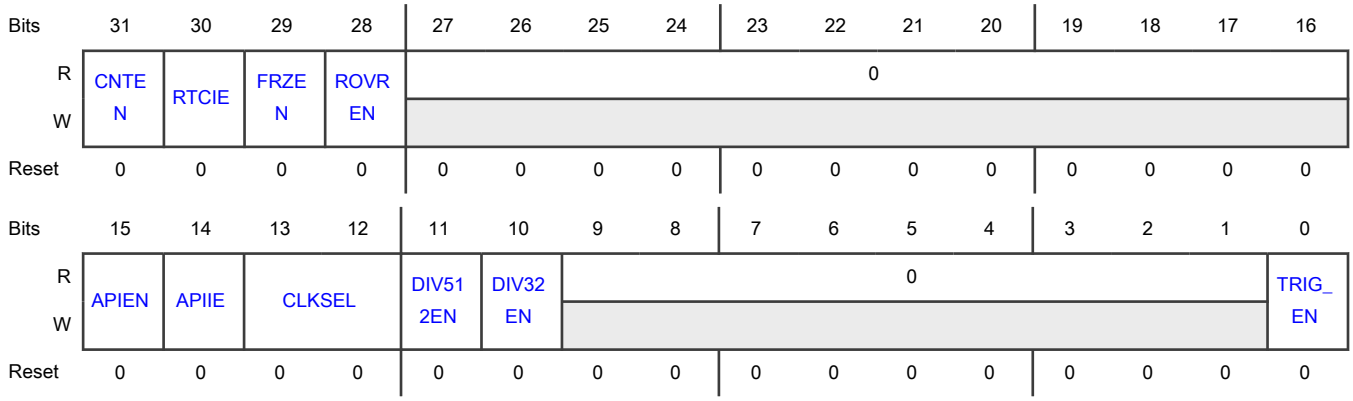
Field	Function
31 SUPV	RTC Supervisor Bit. 0b - All registers are accessible in both user as well as supervisor mode. 1b - All other registers are accessible in the supervisor mode only.
30-0 —	Reserved

66.6.3 RTC Control register (RTCC)

Offset

Register	Offset
RTCC	4h

Diagram



Fields

Field	Function
31 CNTEN	Counter Enable The CNTEN bit enables the RTC counter. Setting CNTEN bit to 0b has the effect of asynchronously resetting (synchronous reset negation) all the RTC and API logic. This allows RTC configuration and clock source selection to be updated without causing synchronization issues. **NOTE** CNTEN should be disabled when INV_RTC, INV_API are cleared. 0b - Counter disabled 1b - Counter enabled
30 RTCIE	RTC Interrupt Enable. The RTCIE bit enables interrupts requests to the system if RTCF is asserted. 0b - RTC interrupts disabled. 1b - RTC interrupts enabled.
29 FRZEN	Freeze Enable Bit The counter freezes on entering the debug mode on the last valid count value if the FRZEN bit is set. After passing of the debug mode counter starts from the frozen value. This bit should not be changed when debug mode is enabled. 0b - Counter does not freeze in debug mode. 1b - Counter freezes in debug mode.
28 ROVREN	Counter Roll Over wakeup/Interrupt Enable. The ROVREN bit enables wakeup and interrupt requests when the RTC has rolled over from 0xFFFF_FFFF to 0x0000_0000. The RTCIE bit must also be set in order to generate an interrupt from a counter rollover. 0b - RTC rollover wakeup/interrupt disabled 1b - RTC rollover wakeup/interrupt enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-16 —	Reserved
15 APIEN	Autonomous Periodic Interrupt Enable. The APIEN bit enables the autonomous periodic interrupt function. Setting this bit to 0b, asynchronously disables API wakeup output of RTC as well. 0b - API disabled. 1b - API enabled.
14 APIIE	API Interrupt Enable. The APIIE bit enables interrupts requests to the system if APIF is asserted. 0b - API interrupts disabled. 1b - API interrupts enabled.
13-12 CLKSEL	Clock select The CLKSEL[1:0] bits select the clock source for the RTC. CLKSEL may only be updated when CNTEN is 0. The user should ensure that oscillator is enabled before selecting it as a clock source for RTC. **NOTE** Refer the RTC clocking section in the Clocking chapter of this document. 00b - Clock source 0 01b - Clock source 1 10b - Clock source 2 11b - Clock source 3
11 DIV512EN	Divide by 512 enable The DIV512EN bit enables the 512 clock divider. DIV512EN may only be updated when CNTEN is 0. 0b - Divide by 512 is disabled. 1b - Divide by 512 is enabled.
10 DIV32EN	Divide by 32 enable. The DIV32EN bit enables the 32 clock divider. DIV32EN may only be updated when CNTEN is 0. 0b - Divide by 32 is disabled 1b - Divide by 32 is enabled
9-1 —	Reserved
0 TRIG_EN	Trigger enable for Analog Comparator

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit (TRIG_EN) when set will de-assert the wakeup_api on the next RTC clock (required for ACMP when bus clock is disabled). **NOTE** If API wakeup is asserted with TRIG_EN set, and bus clock enabled, setting of RTCS[APIF] may be missed.

66.6.4 RTC Status register (RTCS)

Offset

Register	Offset
RTCS	8h

Function

NOTE

W1C has priority over setting of the RTCF, APIF and ROVRF bits, in case both clearing and setting occurs at the same time.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		RTCF	0										INV_RTC	INV_API	Reserved
W			W1C													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		APIF	0		ROVRF	0									
W			W1C			W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29	RTC Interrupt Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
RTCF	The RTCF bit indicates that the RTC counter has reached the counter value matching RTC Compare value register (RTCVAL). RTCF is cleared by writing a 1 to RTCF. Writing a 0 to RTCF has no effect. 0b - RTC counter is not equal to RTCVAL. 1b - RTC counter matches RTCVAL.
28-19 —	Reserved
18 INV_RTC	Invalid RTC write This bit returns value 1 after a value is written to the RTCVAL register and the synchronization process is in progress. During this synchronization period, any attempt to write to the RTCVAL register again is ignored. Synchronization will complete only when CNTEN is set.
17 INV_API	Invalid APIVAL write This bit returns value 1 after a value is written to the APIVAL register and the synchronization process is in progress. During this synchronization period, any attempt to write to the APIVAL register again is ignored. Synchronization will complete only when CNTEN is set.
16 —	Reserved
15-14 —	Reserved
13 APIF	API Interrupt Flag. The APIF bit indicates that the RTC counter has reached the counter value matching API offset value. APIF is cleared by writing a 1 to APIF. Writing a 0 to APIF has no effect. 0b - Counter is not equal to API offset value. 1b - Counter matches the API offset value.
12-11 —	Reserved
10 ROVRF	Counter Roll Over Interrupt Flag. The ROVRF bit indicates that the RTC has rolled over from 0xFFFF_FFFF to 0x0000_0000. ROVRF is cleared by writing a 1 to ROVRF. 0b - RTC has not rolled over 1b - RTC has rolled over
9-0 —	Reserved

66.6.5 RTC Counter register (RTCCNT)

Offset

Register	Offset
RTCCNT	Ch

Diagram

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

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

Fields

Field	Function
31-0 RTCCNT	RTC Counter Value Because of clock synchronization, the RTCCNT value may represent a previous counter value.

66.6.6 API Compare value register (APIVAL)

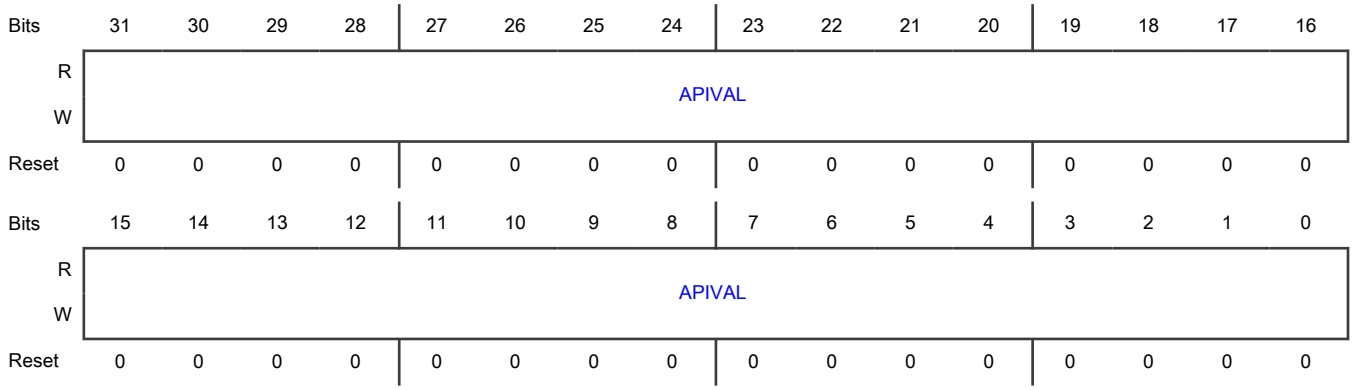
Offset

Register	Offset
APIVAL	10h

Function

The APIVAL offset bits are compared to the RTC counter bits and if a match occurs, an interrupt/wakeup request is asserted.

Diagram



Fields

Field	Function
31-0 APIVAL	API Compare Value. APIVAL bits are added to the current count to calculate an offset. The APIVAL offset bits are compared to the RTC counter bits and if a match occurs, an interrupt/wakeup request is asserted. **NOTE** API functionality is active only when APIVAL is non zero. The first API interrupt takes two more cycles because of synchronization of APIVAL to RTC clock. After that, interrupts are periodic in nature and it takes APIVAL+1 cycles. The Minimum supported value of APIVAL is 4. This is because of synchronization.

66.6.7 RTC Compare value register (RTCVAL)

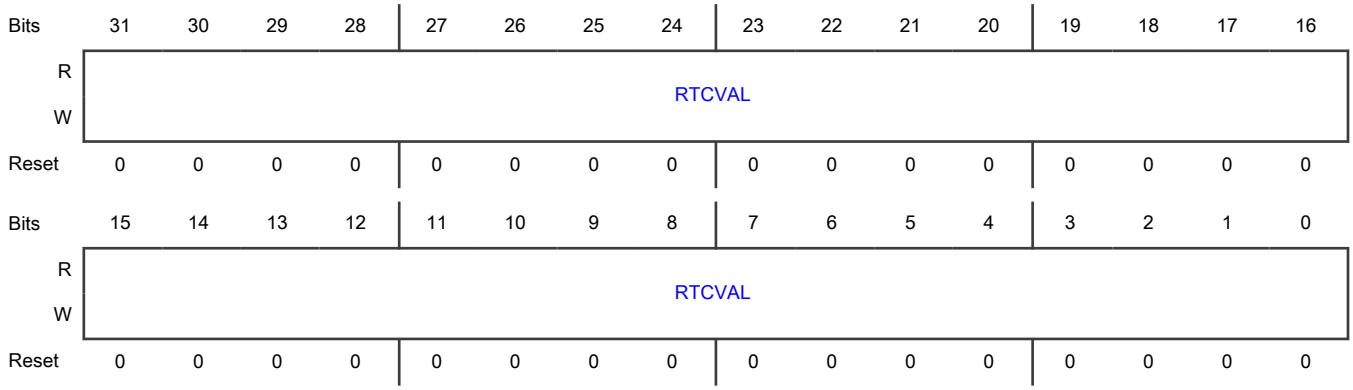
Offset

Register	Offset
RTCVAL	14h

Function

The RTCVAL bits are compared to the RTC counter bits and if a match occurs, RTCF is set. The minimum value of RTCVAL should be 4.

Diagram



Fields

Field	Function
31-0 RTCVAL	RTC Compare Value. The RTCVAL bits are compared to the RTC counter bits and if a match occurs, RTCF is set.

66.7 Glossary

API	Autonomous Periodic Interrupt
IRCs	Internal RC Oscillator
OSCs	Oscillator

Chapter 67
Low Power Serial Peripheral Interface (LPSPI)

67.1 Chip-specific LPSPI information

67.1.1 LPSPI instances and configuration

Table 361. LPSPI instances

Instance	MWCT2D17S	MWCT2015S/MWCT2016S/MWCT2D16S
LPSPI0	Yes	Yes
LPSPI1	Yes	Yes
LPSPI2	Yes	Yes
LPSPI3	Yes	Yes
LPSPI4	Yes	No
LPSPI5	Yes	No

Table 362. LPSPI instances configuration

Instances	TX FIFO Size	RX FIFO Size	Chip Selects
LPSPI0	4x32 bit	4x32 bit	8
LPSPI1	4x32 bit	4x32 bit	6
LPSPI2	4x32 bit	4x32 bit	4
LPSPI3	4x32 bit	4x32 bit	4
LPSPI4	4x32 bit	4x32 bit	4
LPSPI5	4x32 bit	4x32 bit	4

- For supported data rates see table Peripheral data rates in Clocking chapter
- Low leakage and Wait modes are not supported in this device.

The number of chip selects that each LPSPI instance supports varies. Because of that, in the Configuration Register 1 (CFGR1), the Peripheral Chip Select (PCS) field and the Peripheral Chip Select Polarity (PCSPOL) field also vary. See the next 2 tables.

Table 363. LPSPI instances mapped against Peripheral Chip Select (PCS) field support

PCS supported in	PCS not supported in
LPSPI0_TCR[26-24]	—
LPSPI1_TCR[26-24]	—
LPSPI2_TCR[25–24]	LPSPI2_TCR[26]
LPSPI3_TCR[25–24]	LPSPI3_TCR[26]
LPSPI4_TCR[25–24]	LPSPI4_TCR[26]
LPSPI5_TCR[25–24]	LPSPI5_TCR[26]

Table 364. LPSPI instances mapped against Peripheral Chip Select Polarity (PCSPOL) field support

PCSPOL supported in	PCSPOL not supported in
LPSPI0_CFGR1[15-8]	—
LPSPI1_CFGR1[13–8]	LPSPI1_CFGR1[15–14]
LPSPI2_CFGR1[11–8]	LPSPI2_CFGR1[15–12]
LPSPI3_CFGR1[11–8]	LPSPI3_CFGR1[15–12]
LPSPI4_CFGR1[11–8]	LPSPI4_CFGR1[15–12]
LPSPI5_CFGR1[11–8]	LPSPI5_CFGR1[15–12]

67.1.2 LPSPI HREQ considerations for MWCT2D17S

It is recommended that the HREQ pin (when PCS[1] is used as HREQ) should get de-asserted before the completion of frame transfer. In case if the HREQ state is still asserted and LPSPI returns to idle state after frame transfer completion, the HREQ state is internally latched and the next data is written to the transmit FIFO without waiting for the next HREQ assertion.

This limitation is present only while using PCS[1] as HREQ and HREQ remains asserted throughout frame transfer. In case of using trigger input as HREQ, there is no issue.

67.2 Overview

LPSPI is a low-power Serial Peripheral Interface (SPI) module that supports an efficient interface to an SPI bus, either as a master and/or as a slave. The SPI bus is a synchronous serial communication interface used in embedded systems, typically to perform short distance communications between microcontrollers and peripheral devices, on printed circuit boards. Typical applications include interfacing to Secure Digital cards and LCD displays.

67.2.1 Block diagram

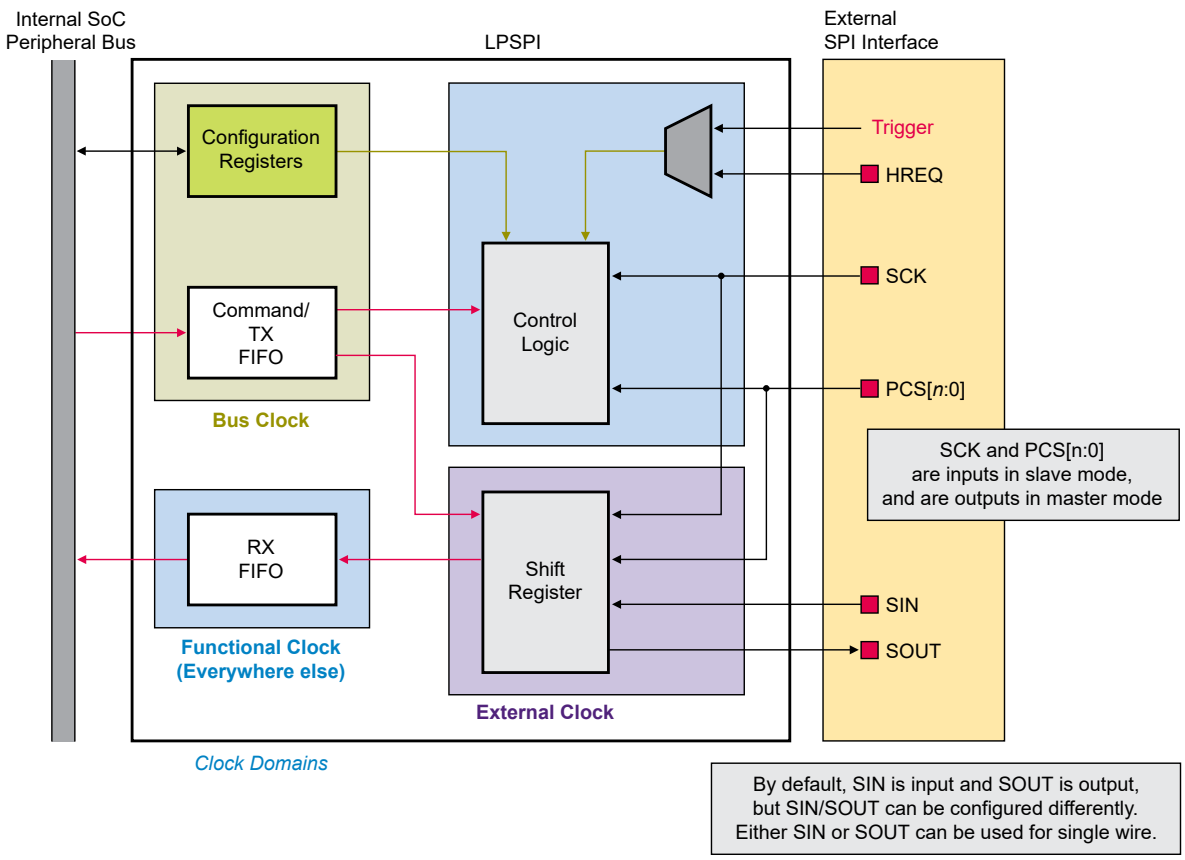


Figure 361. Block diagram

67.2.2 Features

LPSPI supports:

- Uses little CPU overhead, with DMA offloading of FIFO register accesses
- Supports DMA accesses and generates a DMA request
- Word size of 32 bits
- Configurable clock polarity and clock phase
- Master operation supporting up to 8 peripheral chip selects
- Slave operation
- Command/transmit FIFO of 4 words
- Receive FIFO of 4 words
- Flexible timing parameters in Master mode, including SCK frequency, duty cycle, and delays between PCS and SCK edges
- Continuous transfer option to keep the PCS asserted across multiple frames
- Full duplex transfers supporting 1-bit transmit and receive on each clock edge
- Half duplex transfers supporting 1-bit transmit or receive on each clock edge

- Half duplex transfers supporting 2-bit transmit or receive on each clock edge
- Half duplex transfers supporting 4-bit transmit or receive on each clock edge
- Half duplex transfers supporting 8-bit transmit or receive on each clock edge
- Host request can be used to control the start of a SPI bus transfer
- Receive data match logic supporting wakeup on data match

67.3 Functional description

67.3.1 Master mode

67.3.1.1 Transmit and Command FIFO commands

The transmit and command FIFO is a combined FIFO that includes both transmit data words and command words.

- Transmit data words are stored to the transmit/command FIFO, by writing the Transmit Data (TDR).
- Command words are stored to the transmit/command FIFO, by writing the Transmit Command (TCR).

When a command word is at the top of the transmit/command FIFO, the actions that can occur depend upon whether the LPSPI module is either busy or between frames. See Continuous Transfer bit, TCR[CONT] and Continuing Command bit, TCR[CONTC]

Table 365. Possible actions when a command word is at the top of the transmit/command FIFO

Condition	Action
If LPSPI is between frames	then the command word is pulled from the FIFO, and that command word controls all subsequent transfers.
If LPSPI is busy and the Continuing Command bit (TCR[CONTC]) is cleared	then the SPI frame completes at the end of the existing word, ignoring the FRAMESZ configuration. The command word is then pulled from the FIFO and that command word controls all subsequent transfers (or until the next update to the command word). Note that a command word with TCR[CONTC] = 0 always terminates (stop) the existing transfer regardless of the previous TCR[CONT] value.
If LPSPI is busy and the existing TCR[CONT] bit is set and the new TCR[CONTC] value is set	then the command word must be updated at the frame boundary. The command word is pulled from the FIFO during the last SCK pulse of the existing frame (based on the FRAMESZ configuration), and the frame continues using the new command value for the rest of the frame (or until the next update to the command word). When TCR[CONTC] is set, only the lower 24-bits of the command word are updated. If the command word is updated at a word boundary, then the transfer halts (stops) after that word. Essentially the TCR[CONTC] bit is ignored when not at a frame boundary, so the frame ends prematurely.

About TCR[CONT] and TCR[CONTC]:

- TCR[CONT] = 1 is used to keep PCS asserted at end of frame, allowing the transfer to continue.
- TCR[CONTC] = 1 is used to indicate that this command word should not terminate the existing frame, and the transfer can continue using the new command word.

TCR[CONTC] = 1 is restricted in the sense that the new command must load on a frame boundary, and the only way for a transfer to continue from a frame boundary, is when the previous command has TCR[CONT] = 1.

The current state of the existing command word is read from Transmit Command (TCR). It requires at least 3 LPSPI functional clock cycles for TCR to update after TCR is written (assuming an empty FIFO) and LPSPI must be enabled (CR[MEN] bit is set).

Writing the TCR does not initiate a SPI bus transfer, unless the TCR[TXMSK] = 1. When the TCR[TXMSK] bit is set, a new command word is not be loaded until the end of the existing frame (based on FRAMESZ configuration); at the end of the transfer, the TCR[TXMSK] bit clears.

In master mode, the LPSPI command word in TCR controls SPI attributes (using bits and fields in registers).

Table 366. LPSPI Command Word in Master mode

Transmit Command (TCR)		Description	Can this field be modified during a data transfer?
Field	Name		
CPOL	Clock Polarity	Configures the polarity of the SCK pin. Any change of CPOL value will cause a transition on the SCK pin.	N
CPHA	Clock Phase	Configures the clock phase of the transfer.	N
PRESCALE	Prescaler Value	Configures a prescaler used to divide the LPSPI functional clock, to generate the timing parameters of the SPI bus transfer. Changing PRESCALE in conjunction with PCS enables the LPSPI module to connect to different slave devices at different frequencies.	N
PCS	Peripheral Chip Select	Configures which PCS asserts for the transfer; the polarity of PCS is static and configured by CFGR1[PCSPOL]. If CFGR1[PCSCFG] = 1. then PCS[3:2] should not be selected.	N
LSBF	LSB First	Configures if LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted or received first.	Y
BYSW	Byte Swap	Enables byte swap on each 32-bit word when transmitting and receiving data. Byte swapping can be useful when interfacing to devices that organize data as big-endian.	Y
CONT	Continuous Transfer	Configures for a continuous transfer that keeps PCS asserted between frames (as configured by FRAMESZ). A new command word is required to cause PCS to negate. Also supports changing the command word at the frame size boundaries.	Y
CONTC	Continuing Command	Indicates that this is a new command word for the existing continuous transfer. The CONTC bit when set must only be written to the transmit/ command FIFO on a frame boundary.	Y
RXMSK	Receive Data Mask	Masks the receive data and does not store the masked receive data to the receive FIFO or perform receive data matching. Useful for half-duplex transfers or to configure which fields are compared during receive data matching.	Y
TXMSK	Transmit Data Mask	Masks the transmit data, so that masked transmit data is not pulled from transmit FIFO, and the output data pin is tristated (unless configured by CFGR1[OUTCFG]). Useful for half-duplex transfers.	Y
WIDTH	Transfer Width	Configures the number of bits shifted on each SCK pulse. • 1-bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. • 2-bit and 4-bit half-duplex transfers are useful for interfacing to QuadSPI memory devices, and at least one bit (TCR[TXMSK] or TCR[RXMSK]) must also be set.	Y

Table continues on the next page...

Table 366. LPSPI Command Word in Master mode (continued)

Transmit Command (TCR)		Description	Can this field be modified during a data transfer?
Field	Name		
		• 8-bit half-duplex transfers are useful for interfacing to OctelSPI memory devices, and at least one bit (TCR[TXMSK] or TCR[RXMSK]) must also be set.	
FRAMESZ	Frame Size	Configures the frame size in number of bits equal to (FRAMESZ + 1). • The minimum frame size is 8 bits, or 16 bits for an 8-bit transfer. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32-bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO will contain the remainder bits. For example, a 72-bit transfer will consist of 3 words: the 1st and 2nd words are 32-bit, and the 3rd word is 8-bit.	Y

SPI bus transfers:

- LPSPI initiates a SPI bus transfer when all conditions are true:
  - Data is written to the transmit FIFO
  - And the HREQ pin is asserted (or the HREQ function is disabled)
  - And LPSPI is enabled
- To perform the SPI bus transfer, LPSPI uses the attributes configured in Transmit Command (TCR) and uses the timing parameters in Clock Configuration (CCR).
- The SPI bus transfer ends after the FRAMESZ configuration is reached (provided CONT=0), or at the end of a word when a new transmit command word is at the top of the transmit/command FIFO. When LPSPI is disabled, the SPI bus transfers end after the transmit FIFO is empty and LPSPI is idle.
- The HREQ input is only checked when PCS is negated.

Circular FIFO: The transmit/command FIFO also supports a Circular FIFO feature, which enables the LPSPI master to (periodically) repeat a short data transfer that fits within the transmit/command FIFO, without requiring additional FIFO accesses. When the circular FIFO is enabled (CFGR0[CIRFIFO] = 1), the current state of the FIFO read pointer is saved and the status flags do not update. After the transmit/command FIFO is considered empty and LPSPI is idle, the FIFO read pointer is restored with the saved version, so the contents of the transmit/command FIFO are not permanently pulled from the FIFO while circular FIFO mode is enabled.

67.3.1.2 Receive FIFO and Data Match

The receive FIFO stores receive data during SPI bus transfers. When TCR[RXMSK] = 1, the receive data is discarded instead of being stored in the receive FIFO.

- The receive data is written to the receive FIFO when the last bit of the word is sampled.
- If the transmit FIFO is empty during a multiple word or continuous transfer, then the receive data is written to the receive FIFO before the transfer stalls (assuming CFGR1[NOSTALL] is clear) while waiting for new transmit data or command word to be written.

Receive data supports a receive data match function that can match received data against one of two words in the DMR0 and DMR1 registers or against a masked data word. The receive data match function can also be configured to compare only the first one or two received data words since the start of the frame.

- Received data that is already discarded due to the TCR[RXMSK] bit cannot cause the data match to set, and delays the receive data match on the first received data word, until all discarded data is received.
- The receive data match function can also be configured to discard all received data until a data match is detected, using the CFGR0[RDMO] bit.
- After a receive data match, to allow all subsequent data to be received, clear the CFGR0[RDMO] bit, then clear the SR[DMF] bit.

67.3.1.3 Timing parameters

The timing parameters that are used for all SPI bus transfers are relative to the LPSPI functional clock divided by the PRESCALE configuration. Although the Clock Configuration (CCR) cannot be changed when the LPSPI module is busy, to support interfacing to different slave devices at different frequencies, the TCR[PRESCALE] configuration can be changed between SPI bus transfers using the Transmit Command Register (TCR).

NOTE

The minimum value below is the minimum counter value, but the clock configuration register values must also satisfy the datasheet specs based on the LPSPI functional clock frequency and prescaler value.

Table 367. LPSPI timing parameters

Clock Configuration (CCR) Clock Configuration 1 (CCR1)		Description	Min	Max
Field	Name			
SCKSET	SCK Setup Phase	Configures the SCK setup phase to (SCKSET + 1) cycles. The setup phase is the SCK high period when either CPHA = 0, CPOL = 1 or CPHA = 1, CPOL = 0, otherwise is it the SCK low period. The SCK period is defined as (SCKSET + SCKHLD + 2) and the duty cycle is the difference between SCKSET and SCKHLD.	0 (1 cycle)	255 (256 cycles)
SCKHLD	SCK Hold Phase	Configures the SCK hold phase to (SCKHLD + 1) cycles. The hold phase is the SCK low period when either CPHA = 0, CPOL = 1 or CPHA = 1, CPOL = 0, otherwise it is the SCK high period. The SCK period is defined as (SCKSET + SCKHLD + 2) and the duty cycle is the difference between SCKSET and SCKHLD.	0 (1 cycle)	255 (256 cycles)
PCSPCS	PCS to PCS Delay	Configures the minimum delay between PCS negation and the next PCS assertion to (PCSPCS + PCSPCS + 2) cycles. When the command word is updated between transfers, there is a minimum of (PCSPCS + 1) cycles between the command word update and any change on PCS pins.	0 (2 cycles)	255 (512 cycles)
SCKSCK	SCK to SCK Delay	Configures the delay during a continuous transfer between the last SCK edge of a frame and the first SCK edge of the continuing frame to (SCKSCK +	0 (1 cycle)	255 (256 cycles)

Table continues on the next page...

Table 367. LPSPI timing parameters (continued)

Clock Configuration (CCR) Clock Configuration 1 (CCR1)		Description	Min	Max
Field	Name			
		1) cycles. This is useful when the external slave requires a large delay between different words of an SPI bus transfer.		
PCSSCK	PCS-to-SCK Delay	Configures the minimum delay between PCS assertion and the first SCK edge to (PCSSCK + 1) cycles.	0 (1 cycle)	255 (256 cycles)
SCKPCS	SCK-to-PCS Delay	Configures the minimum delay between the last SCK edge and the PCS assertion to (SCKPCS + 1) cycles.	0 (1 cycle)	255 (256 cycles)

This figure shows the timing settings controlled by TCR[CPHA], TCR[CPOL], CCR[SCKPCS], CCR[PCSSCK], CCR1[SCKSET], and CCR1[SCKHLD].

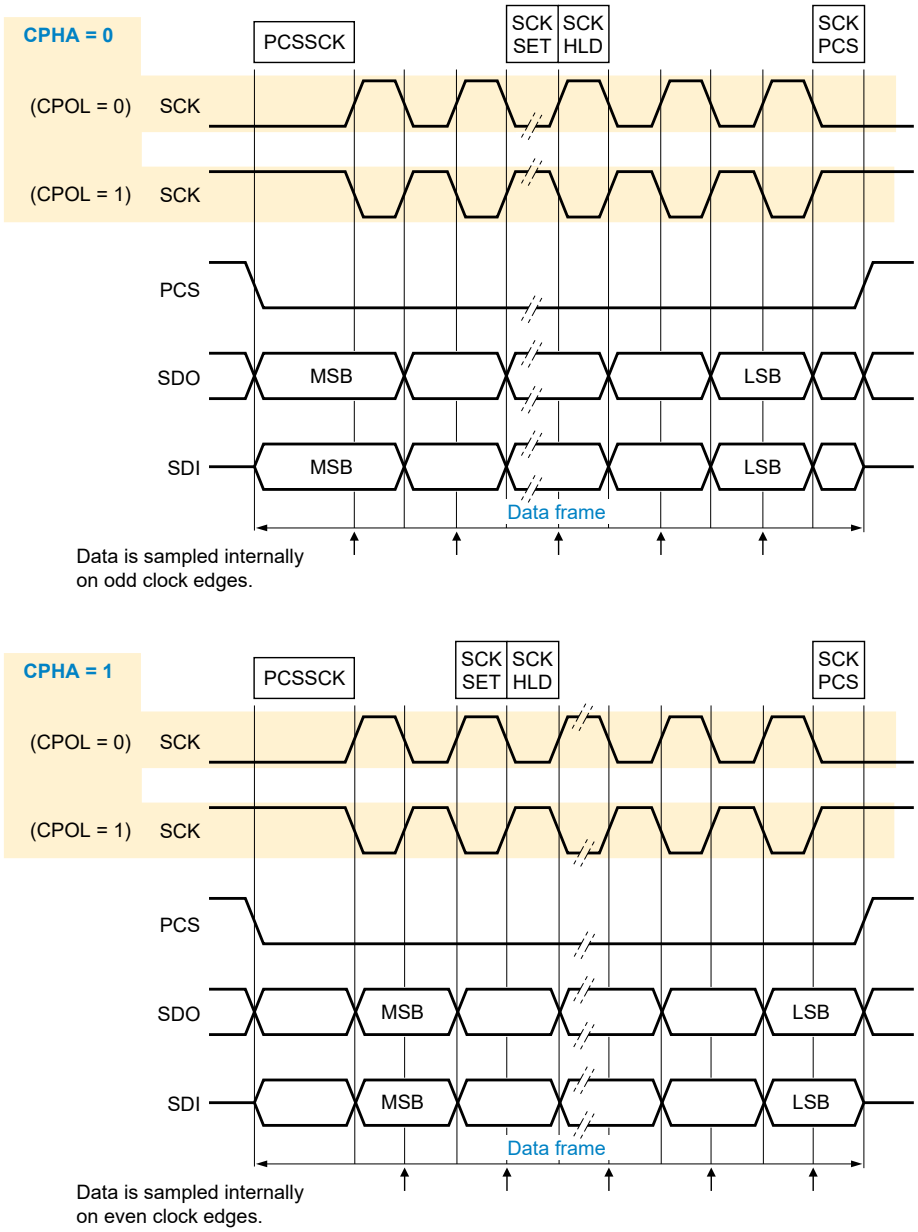


Figure 362. Clock Phase (TCR[CPHA]) timing diagram example

67.3.1.4 Pin configuration

- To swap directions or support half-duplex transfers on the same pin, the SIN and SOUT pins can be configured using CFGR1[PINCFG].
- To determine if an output data pin (like SOUT) tristates when PCS is negated, or if the output data pin simply retains the last value, use CFGR1[PCSCFG].
- When configuring for half-duplex transfers the output data pins must be configured to tristate when PCS is negated; use CFGR1[OUTCFG] CFGR1[OUTCFG].
- When performing half-duplex 2-bit transfers, CFGR1[PCSCFG] can be set to any value.
- When performing half-duplex 4-bit transfers, CFGR1[PCSCFG] must equal 0x1.

67.3.1.5 Clock loopback

Configure the LPSPI master to use either the output clock or a loopback clock to sample the receive data (for example, SIN). Use one of these these clocks to sample the input data:

- The SCK output clock directly
- A delayed version of the SCK output clock

The delayed version of the SCK is chosen by the SCK pin output delay, plus the SCK pin input delay, and is configured by setting CFGR1[SAMPLE]. Enabling the loopback version of the SCK can improve the setup time of the input data from the slave.

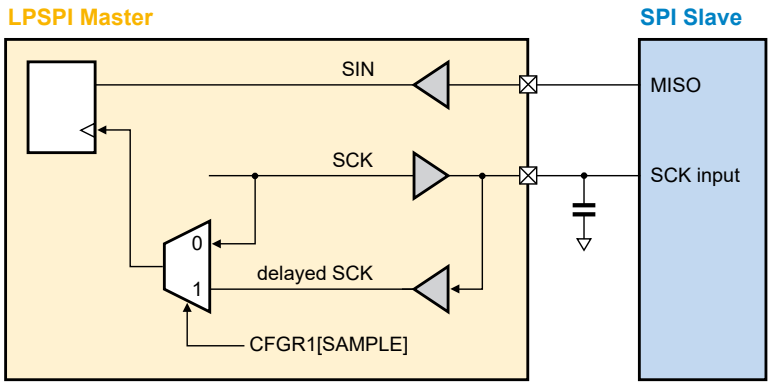


Figure 363. Clock loopback

See the chip data sheet for the specific input setup time in master loopback mode.

67.3.2 Slave mode

LPSPI slave mode:

- Uses the same shift register and logic that master mode uses
- Does not use the Clock Configuration Register (CCR)
- During SPI bus transfers, requires that the Transmit Command (TCR) register remain static (unchanging)

67.3.2.1 Transmit and Command FIFO commands

Before enabling LPSPI in slave mode, initialize the Transmit Command (TCR) register, although the TCR register does not update until after LPSPI is enabled. After being enabled, the TCR register should only be changed if LPSPI is idle. In slave mode, the LPSPI command word in TCR controls SPI attributes. Before the PCS input asserts, the transmit FIFO must be filled with transmit data, or the transmit error flag sets.

Table 368. LPSPI Command Word in Slave mode

Transmit Command (TCR)		Description
Field	Name	
CPOL	Clock Polarity	Configures the polarity of the external SCK input.
CPHA	Clock Phase	Configures the clock phase of transfer.
PRESCALE	Prescaler Value	Configures the LPSPI functional clock prescaler.

Table continues on the next page...

Table 368. LPSPI Command Word in Slave mode (continued)

Transmit Command (TCR)		Description
Field	Name	
PCS	Peripheral Chip Select	Configures which PCS is used, the polarity of PCS is static and configured by CFGR1[PCSPOL]. If CFGR1[PCSCFG] is not equal to zero, then PCS[3:2] pins should not be selected.
LSBF	LSB First	Configures if LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted or received first.
BYSW	Byte Swap	Enables byte swap on each 32-bit word when transmitting and receiving data. Byte swapping can be useful when interfacing to devices that organize data as big-endian.
CONT	Continuous Transfer	When Continuous Transfer is set in slave mode and after the first FRAMESZ bits are transferred, the LPSPI will passthrough and transmit the received data until the next PCS negation. Whatever is shifted in on the receive data is shifted out as transmit data as if there was a 32-bit shift register.
CONTC	Continuing Command	When Continuing Command is set in slave mode and after the first FRAMESZ bits are transferred, then RXMSK will be considered set and TXMSK will be considered clear until the next PCS negation. CONTC can be used to flip the direction of a transfer after the first FRAMESZ bits.
RXMSK	Receive Data Mask	Masks the receive data and does not store the masked receive data to the receive FIFO or perform receive data matching. Useful for half-duplex transfers or to configure which fields are compared during receive data matching.
TXMSK	Transmit Data Mask	Masks the transmit data, so that the masked transmit data is not pulled from transmit FIFO, and the output data pin is tristated (unless configured by CFGR1[OUTCFG]). Useful for half-duplex transfers.
WIDTH	Transfer Width	Configures the number of bits shifted on each SCK pulse. • 1-bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. • 2-bit and 4-bit half-duplex transfers are useful for interfacing to QuadSPI memory devices, and at least one bit (Transmit Data Mask (TCR[TXMSK] or Receive Data Mask TCR[RXMSK]) must also be set. • 8-bit half-duplex transfers are useful for interfacing to OctelSPI memory devices, and at least one bit (Transmit Data Mask (TCR[TXMSK] or Receive Data Mask TCR[RXMSK]) must also be set.
FRAMESZ	Frame Size	Configures the frame size in number of bits equal to (FRAMESZ + 1). • The minimum frame size is 8 bits, or 16 bits for an 8-bit parallel transfer. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32-bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO will contain the remainder bits. For example, a 72-bit transfer will consist of 3 words: the 1st and 2nd words are 32-bit, and the 3rd word is 8-bit.

67.3.2.2 Receive FIFO and data match

The receive FIFO stores receive data during SPI bus transfers. When TCR[RXMSK] = 1, the received data is discarded instead of storing the received data in the receive FIFO.

Receive data supports a receive data match function that can match received data against one of two words in the DMR0 and DMR1 registers or against a masked data word. The data match function can also be configured to compare only the first one or two received data words since the start of the frame.

- Received data that is already discarded because TCR[RXMSK] = 1 cannot cause the data match to set, and delays the match on the first received data word, until all discarded data is received.
- The receiver match function can also be configured to discard all received data until a data match is detected, using the CFGR0[RDMO] bit.
- After a receive data match, to allow all subsequent data to be received, first clear the CFGR0[RDMO] bit, then clear the SR[DMF] bit.

67.3.2.3 Partial Received Word

When the PCS negates and the receive shift register has shifted in a partial word, the receive shift register can be configured to discard the partial word or to store it in the receive FIFO. This is controlled by the CFGR1[PARTIAL] configuration bit.

A partial word is defined as less than FRAMESZ bits (when FRAMESZ is equal or less than 32 bits, or it is the last word in a multi-word frame) or less than 32 bits (when FRAMESZ is greater than 32 bits and not the last word in a multi-word frame).

A single bit frame is not supported. A partial received word of 1 bit is fine, but a partial received frame of 1 bit is not supported.

67.3.2.4 Clocked interface

The LPSPI module supports interfacing to external masters that provide only clock and data pins (PCS is not required). This interface requires:

- Using Clock Phase TCR[CPHA] = 1 (data is changed on the leading edge of SCK and captured on the following edge)
- Configuring the PCS input to be always asserted (CFGR1[PCSPOLn] = 1). For example, to configure PCS[0] to be always asserted, set PCSPOL[0] = 1, and do not configure PCS[0] in the pin muxing. The chip-level drives PCS to a certain value (ideally 1), CFGR1[PCSPOLn] could be used to invert that value.
- Setting CFGR1[AUTOPCS] = 1 (Automatic PCS generation is enabled). When CFGR1[AUTOPCS] = 1, a minimum of 4 LPSPI functional clock cycles (divided by PRESCALE configuration) is required between the last SCK edge of one word and the first SCK edge of the next word.

67.3.3 Low power modes

Table 369. Low power modes

Chip mode	LPSPI operation
Run	Normal operation

67.3.4 Debug mode

Table 370. Debug mode

Chip mode	LPSPI operation
Debug (the core is in Debug/ Halted mode)	Can continue operating in Debug mode, if the CR[DBGEN] = 1.

67.3.5 Interrupts and DMA Requests

The next table lists the slave mode sources (status flags) that can generate LPSPI interrupts and LPSPI slave transmit/receive DMA requests.

Table 371. LPSPI Interrupts and DMA Requests

Status (SR)		Description	Can generate		
Status Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
TDF	Transmit Data Flag	Data can be written to transmit FIFO, as configured by the Transmit FIFO Watermark FCR[TXWATER]	Y	TX	Y
RDF	Receive Data Flag	Data can be read from the receive FIFO, as configured by the Receive FIFO Watermark FCR[RXWATER]	Y	RX	Y
WCF	Word Complete Flag	Word is complete, the last bit of the word has been sampled	Y	N	Y
FCF	Frame Complete Flag	Frame is complete, and PCS has negated	Y	N	Y
TCF	Transfer Complete Flag	Transfer is complete, PCS has negated, and the transmit/command FIFO is empty	Y	N	Y
TEF	Transmit Error Flag	Indicates a transmit/command FIFO underrun. In master mode when CFGR1[NOSTALL] = 0 (transfers stall when transmit FIFO is empty), the Transmit Error Flag bit cannot set.	Y	N	Y
REF	Receive Error Flag	Receive error flag, indicates a receive FIFO overflow. In master mode when CFGR1[NOSTALL] = 0 (transfers stall when receive FIFO is full), the Receive Error Flag bit cannot set.	Y	N	Y
DMF	Data Match Flag	Indicates that the received data has matched the configured data match value	Y	N	Y
MBF	Module Busy Flag	LPSPI is busy performing a SPI bus transfer	N	N	N

67.3.6 Clocks

Table 372. LPSPI Clocks

LPSPI Functional clock	• The LPSPI functional clock is asynchronous to the bus clock. • If the LPSPI functional clock remains enabled in low power modes, then LPSPI can perform SPI bus transfers and low power wakeups, in both master and slave modes.

Table continues on the next page...

Table 372. LPSPI Clocks (continued)

	• The LPSPI divides the functional clock by a prescaler; the resulting frequency must be at least 2 times faster than the SPI external clock frequency (SCK).
External clock	• The LPSPI shift register is clocked directly by the SCK clock. • How the SCK clock is generated or supplied depends upon the mode (master or slave): — In master mode: the SCK clock is generated internally. — In slave mode: the SCK clock is supplied externally.
Bus clock	The bus clock is only used for bus accesses to the LPSPI control and configuration registers. The bus clock frequency must be high enough to support the data bandwidth requirements of the LPSPI registers, including the FIFOs.

For chip-specific clocking information, see the Clocking chapter.

67.3.7 Resets

Table 373. Resets

Chip reset	Resets the LPSPI logic and registers to their default state.
Software reset	• Resets the LPSPI logic and registers to their default state, except for the Control Register. • The LPSPI software reset is in the Control Register CR[RST].
FIFO resets	• Resets the transmit/command FIFO and the receive FIFO. • The Reset Transmit FIFO field, CR[RTF] and the Reset Receive FIFO field, CR[RRF] are write-only bits. • After being reset, a FIFO is empty.

67.3.8 Peripheral Triggers

The connection of the LPSPI peripheral triggers to other peripherals depend upon the specific device being used.

Table 374. LPSPI Triggers

Trigger	Description	Notes
Frame Output Trigger	The frame output trigger: • asserts at the end of each frame (when PCS negates) • remains asserted for one cycle of LPSPI functional clock divided by the PRESCALE configuration	LPSPI generates 2 output triggers that can be connected to other peripherals on the device.
Word Output Trigger	The word output trigger: • asserts at the end of each received word • remains asserted for one cycle of LPSPI functional clock divided by the PRESCALE configuration	
Input Trigger	To control the start of a LPSPI bus transfer, the LPSPI input trigger can be selected instead of the HREQ input.	

Table continues on the next page...

Table 374. LPSPI Triggers (continued)

Trigger	Description	Notes
	• The LPSPI input trigger is synchronized, and must assert for at least 2 cycles of the LPSPI functional clock divided by the PRESCALE configuration, so that the input trigger can be detected. • When the LPSPI module is busy, the HREQ input (and therefore the LPSPI input trigger) is ignored. • Both HREQ and the LPSPI input trigger are ignored when the module is busy. They are used to start a new transfer when the module is idle.	

67.4 Signal descriptions

Table 375. Signals

Signal	Name	Description	I/O
SCK	Serial clock	• Input in slave mode • Output in master mode	I/O
PCS[0]	Peripheral Chip Select	• Input in slave mode • Output in master mode	I/O
PCS[1] / HREQ	Peripheral Chip Select or Host Request	Host Request pin is selected when CFGR0[HREN] = 1 and CFGR0[HRSEL] = 0 • Input in either slave mode or when used as master Host Request • Output in either master mode or when used as slave Host Request	I/O
PCS[2] / DATA[2]	Peripheral Chip Select or data pin 2 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in slave mode • Output in master mode When CFGR1[PCSCFG] = 1: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
PCS[3] / DATA[3]	Peripheral Chip Select or data pin 3 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in slave mode • Output in master mode When CFGR1[PCSCFG] = 1: • Input in half-duplex parallel data receive transfers	I/O

Table continues on the next page...

Table 375. Signals (continued)

Signal	Name	Description	I/O
		• Output in half-duplex parallel data transmit transfers	
PCS[4] / DATA[4]	Peripheral Chip Select or data pin 4 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in slave mode • Output in master mode When CFGR1[PCSCFG] = 11b: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
PCS[5] / DATA[5]	Peripheral Chip Select or data pin 5 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in slave mode • Output in master mode When CFGR1[PCSCFG] = 11b: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
PCS[6] / DATA[6]	Peripheral Chip Select or data pin 6 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in slave mode • Output in master mode When CFGR1[PCSCFG] = 11b: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
PCS[7] / DATA[7]	Peripheral Chip Select or data pin 7 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in slave mode • Output in master mode When CFGR1[PCSCFG] = 11b: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
SOUT / DATA[0]	Serial Data Output	Can be configured as serial data input signal	I/O

Table continues on the next page...

Table 375. Signals (continued)

Signal	Name	Description	I/O
		• Used as data pin 0 in half-duplex parallel data transfers	
SIN / DATA[1]	Serial Data Input	Can be configured as serial data output signal • Used as data pin 1 in half-duplex parallel data transfers	I/O

67.5 Memory map and registers

NOTE

Writing a Read-Only (RO) register or reading a Write-Only (WO) register can cause bus errors. LPSPI does not check if programmed values in the registers are correct, so application software must take care to write valid values only.

67.5.1 LPSPI register descriptions

67.5.1.1 LPSPI memory map

LPSPI_0 base address: 4035_8000h

LPSPI_1 base address: 4035_C000h

LPSPI_2 base address: 4036_0000h

LPSPI_3 base address: 4036_4000h

LPSPI_4 base address: 404B_C000h

LPSPI_5 base address: 404C_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0200_0004h
4h	Parameter (PARAM)	32	R	See section
10h	Control (CR)	32	RW	0000_0000h
14h	Status (SR)	32	RW	0000_0001h
18h	Interrupt Enable (IER)	32	RW	0000_0000h
1Ch	DMA Enable (DER)	32	RW	0000_0000h
20h	Configuration 0 (CFGR0)	32	RW	0000_0000h
24h	Configuration 1 (CFGR1)	32	RW	0000_0000h
30h	Data Match 0 (DMR0)	32	RW	0000_0000h
34h	Data Match 1 (DMR1)	32	RW	0000_0000h
40h	Clock Configuration (CCR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
44h	Clock Configuration 1 (CCR1)	32	RW	0000_0000h
58h	FIFO Control (FCR)	32	RW	0000_0000h
5Ch	FIFO Status (FSR)	32	R	0000_0000h
60h	Transmit Command (TCR)	32	RW	0000_001Fh
64h	Transmit Data (TDR)	32	W	0000_0000h
70h	Receive Status (RSR)	32	R	0000_0002h
74h	Receive Data (RDR)	32	R	0000_0000h
78h	Receive Data Read Only (RDROR)	32	R	0000_0000h
3FCh	Transmit Command Burst (TCBR)	32	W	0000_0000h
400h - 5FCh	Transmit Data Burst (TDBR0 - TDBR127)	32	W	0000_0000h
600h - 7FCh	Receive Data Burst (RDBR0 - RDBR127)	32	R	0000_0000h

67.5.1.2 Version ID (VERID)

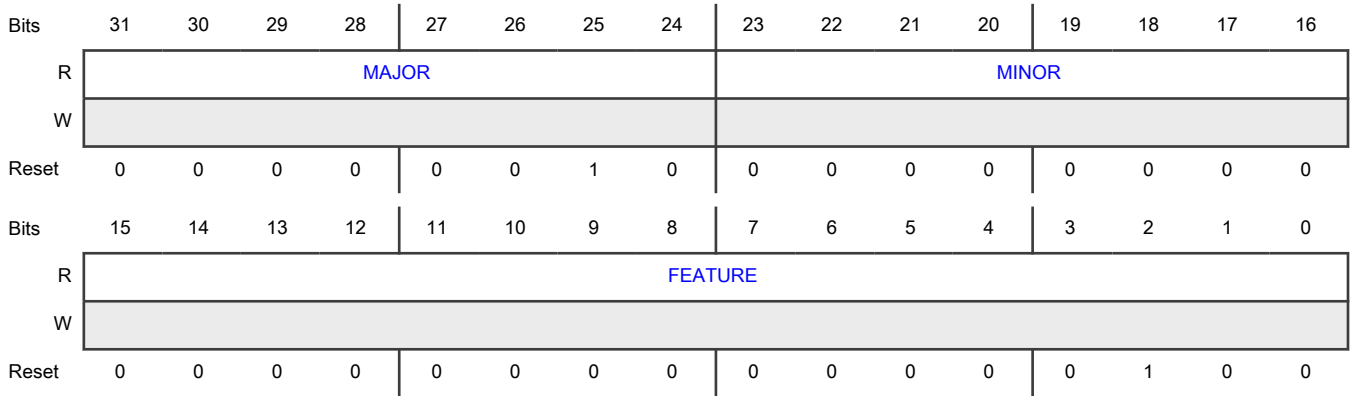
Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module specification. Read-only field.
23-16 MINOR	Minor Version Number Returns the minor version number for the module specification. Read-only field.
15-0 FEATURE	Module Identification Number Returns the feature set number. Read-only field. 0000_0000_0000_0100b - Standard feature set supporting a 32-bit shift register.

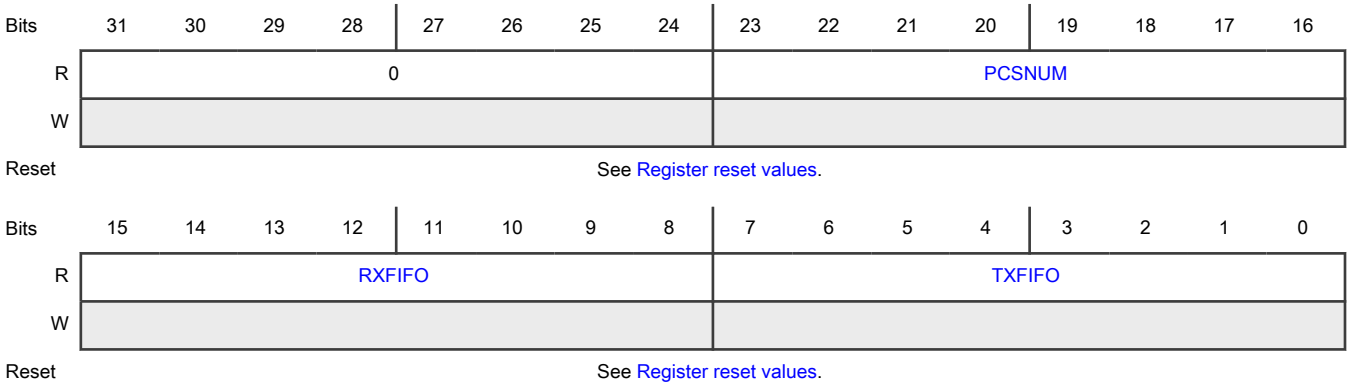
67.5.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function
Contains parameter values that were implemented in the module.

Diagram



Register reset values

Register	Reset value
PARAM	LPSPI_0: 0008_0202h LPSPI_1: 0006_0202h LPSPI_2–LPSPI_5: 0004_0202h

Fields

Field	Function
31-24 —	Reserved
23-16 PCSNUM	PCS Number Sets the number of PCS pins supported by the peripheral.
15-8 RXFIFO	Receive FIFO Size Indicates the maximum number of words in the receive FIFO, which is 2^{RXFIFO}.
7-0 TXFIFO	Transmit FIFO Size Indicates the maximum number of words in the transmit FIFO, which is 2^{TXFIFO}.

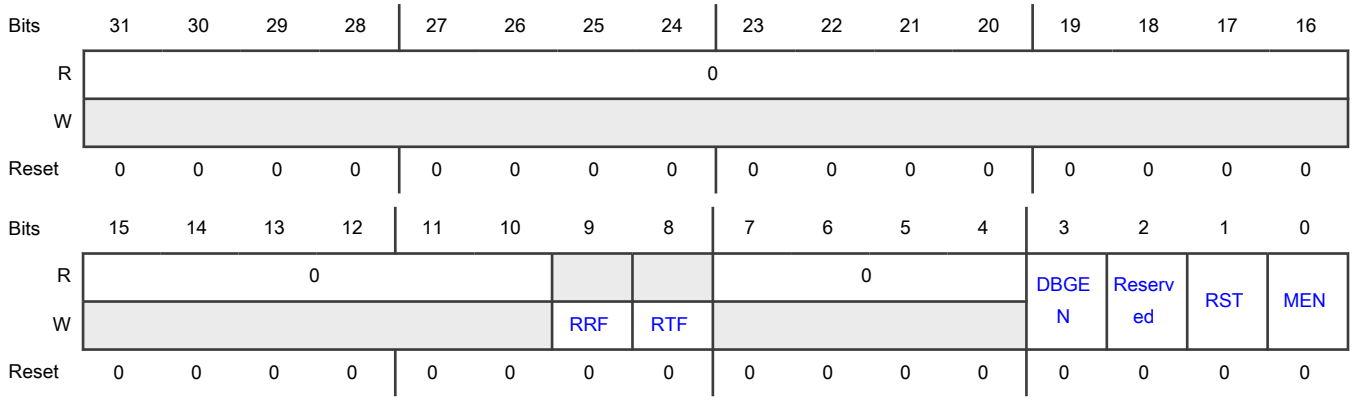
67.5.1.4 Control (CR)

Offset

Register	Offset
CR	10h

Function
Contains fields associated with the module operation.

Diagram



Fields

Field	Function
31-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 RRF	Reset Receive FIFO 0b - No effect 1b - Reset the Receive FIFO. The register bit always reads zero.
8 RTF	Reset Transmit FIFO 0b - No effect 1b - Reset the Transmit FIFO. The register bit always reads zero.
7-4 —	Reserved
3 DBGEN	Debug Enable Enables or disables the LPSPI module in debug mode. Debug Enable bit should be updated only when the LPSPI module is disabled. 0b - LPSPI module is disabled when the CPU is halted. When LPSPI is disabled, the PCS will be negated once the transmit FIFO is empty regardless of the state of TCR register. 1b - LPSPI module is enabled in debug mode
2 —	Reserved
1 RST	Software Reset Reset all internal logic and registers, except the Control Register. RST remains set until cleared by software. The reset takes effect immediately and remains asserted until negated by software. There is no minimum delay required before clearing the software reset. 0b - Module is not reset 1b - Module is reset
0 MEN	Module Enable 0b - Module is disabled 1b - Module is enabled

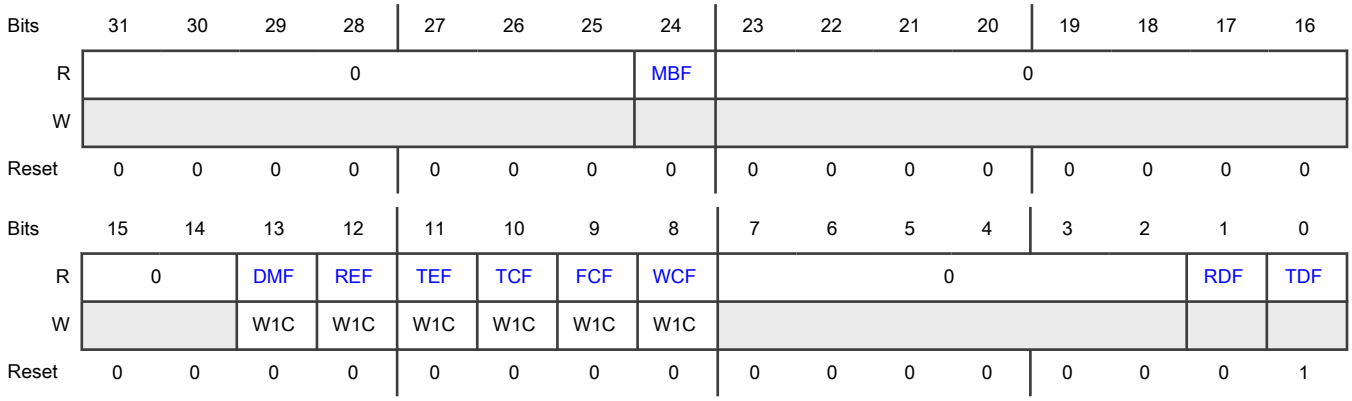
67.5.1.5 Status (SR)

Offset

Register	Offset
SR	14h

Function
Contains the status of data flow.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 MBF	Module Busy Flag In Master mode, MBF asserts when there is data to transmit and LPSPI is able to transmit (eg: HREQ asserted, etc). It negates after the PCS negates and the LPSPI master has waited half the DBT time with no new data to transmit. Slave mode asserts MBF when LPSPI is enabled and PCS is asserted. 0b - LPSPI is idle 1b - LPSPI is busy
23-14 —	Reserved
13 DMF	Data Match Flag Indicates that the received data has matched the DMR0[MATCH0] and/or DMR1[MATCH1] fields (as configured by CFGR1[MATCFG]. 0b - Have not received matching data 1b - Have received matching data
12 REF	Receive Error Flag The Receive Error Flag sets when the Receiver FIFO overflows. When the Receive Error Flag is set, it is recommended to first end the transfer, empty the Receive FIFO, clear the Receive Error Flag and then restart the transfer from the beginning. 0b - Receive FIFO has not overflowed 1b - Receive FIFO has overflowed
11 TEF	Transmit Error Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The Transmit Error Flag sets when the Transmit FIFO underruns. When the Transmit Error Flag is set, it is recommended to first end the transfer, clear the Transmit Error Flag and then restart the transfer from the beginning. 0b - Transmit FIFO underrun has not occurred 1b - Transmit FIFO underrun has occurred
10 TCF	Transfer Complete Flag In Master mode when LPSPI returns to idle state with the transmit FIFO empty, the Transfer Complete Flag sets. 0b - All transfers have not completed 1b - All transfers have completed
9 FCF	Frame Complete Flag The Frame Complete Flag sets at the end of each frame transfer, when the PCS negates. 0b - Frame transfer has not completed 1b - Frame transfer has completed
8 WCF	Word Complete Flag The Word Complete Flag sets when the last bit of a received word is sampled. 0b - Transfer of a received word has not yet completed 1b - Transfer of a received word has completed
7-2 —	Reserved
1 RDF	Receive Data Flag The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than the value in FCR[RXWATER]. 0b - Receive Data is not ready 1b - Receive data is ready
0 TDF	Transmit Data Flag The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than the value in FCR[TXWATER]. 0b - Transmit data not requested 1b - Transmit data is requested

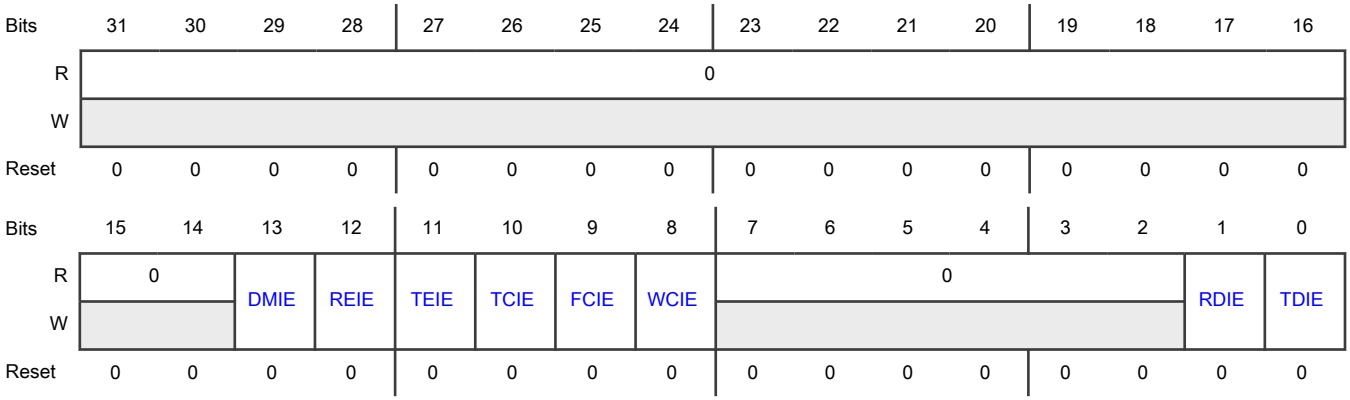
67.5.1.6 Interrupt Enable (IER)

Offset

Register	Offset
IER	18h

Function
Enables interrupts based on data flow and errors.

Diagram



Fields

Field	Function
31-14 —	Reserved
13 DMIE	Data Match Interrupt Enable 0b - Disabled 1b - Enabled
12 REIE	Receive Error Interrupt Enable 0b - Disabled 1b - Enabled
11 TEIE	Transmit Error Interrupt Enable 0b - Disabled 1b - Enabled
10 TCIE	Transfer Complete Interrupt Enable 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 FCIE	Frame Complete Interrupt Enable 0b - Disabled 1b - Enabled
8 WCIE	Word Complete Interrupt Enable 0b - Disabled 1b - Enabled
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable 0b - Disabled 1b - Enabled
0 TDIE	Transmit Data Interrupt Enable 0b - Disabled 1b - Enabled

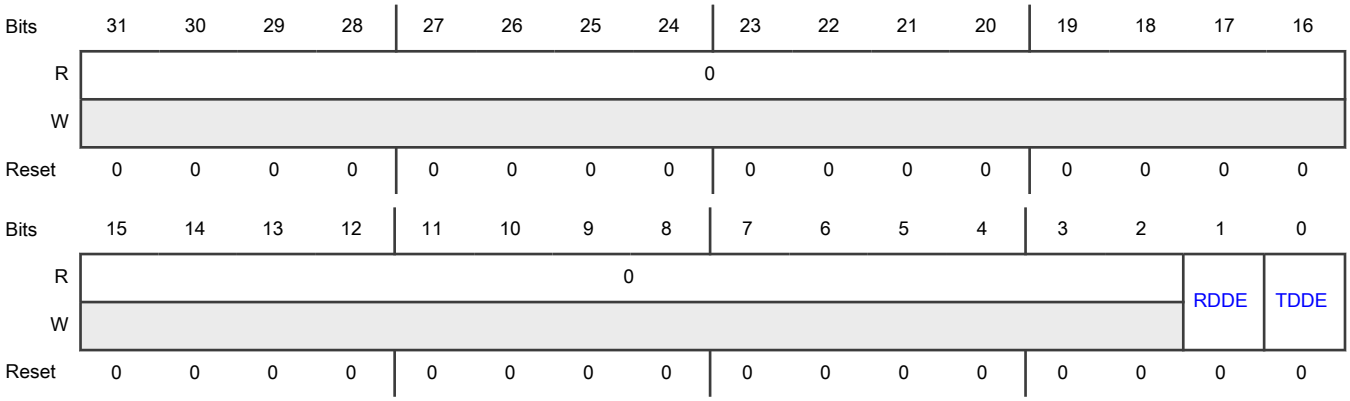
67.5.1.7 DMA Enable (DER)

Offset

Register	Offset
DER	1Ch

Function
Enables DMA data flow.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RDDE	Receive Data DMA Enable 0b - DMA request is disabled 1b - DMA request is enabled
0 TDDE	Transmit Data DMA Enable 0b - DMA request is disabled 1b - DMA request is enabled

67.5.1.8 Configuration 0 (CFGR0)

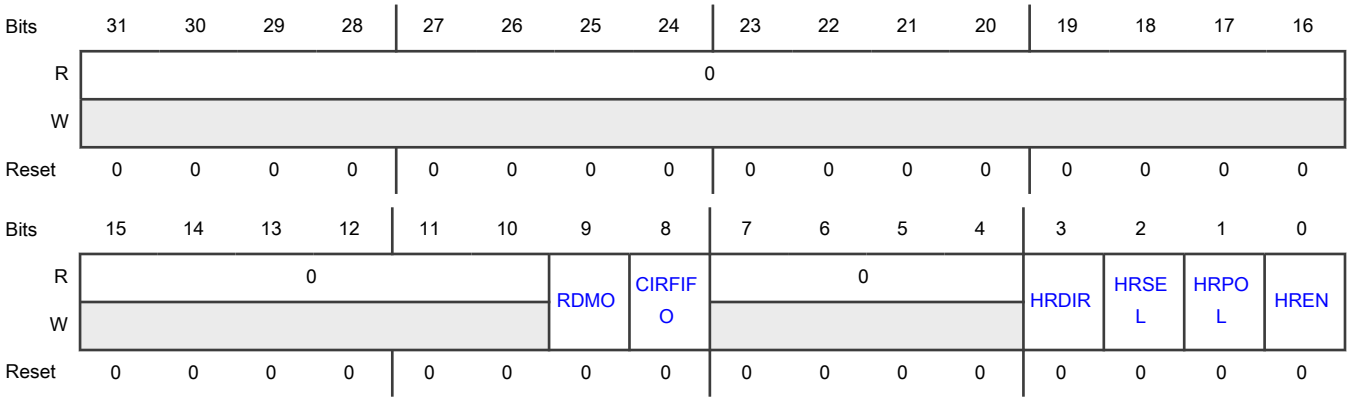
Offset

Register	Offset
CFGR0	20h

Function

Includes additional fields to configure LPSPI including Circular FIFO Enable and Receive Data Match Only.

Diagram



Fields

Field	Function
31-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 RDMO	Receive Data Match Only When RDMO is enabled, all received data that does not cause the Data Match Flag (SR[DMF]) to set is discarded. • Set the RDMO bit when LPSPI is idle and SR[DMF] = 0. • After SR[DMF] = 1, the RDMO bit configuration is ignored. • When disabling RDMO and to ensure that no receive data is lost, before clearing SR[DMF], first clear RDMO. 0b - Received data is stored in the receive FIFO as in normal operations 1b - Received data is discarded unless the SR[DMF] = 1
8 CIRFIFO	Circular FIFO Enable When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO is emptied as it normally is, but after LPSPI is idle and the transmit FIFO is empty, then the read pointer value is restored from the temporary register. This restoring of the read pointer causes the contents of the transmit FIFO to be cycled through repeatedly. **NOTE** The read pointer is restored for as long as the CIRFIFO is set. Writing additional words to the FIFO after setting CIRFIFO adds them to the end of the FIFO, up to the size of the transmit FIFO. 0b - Circular FIFO is disabled 1b - Circular FIFO is enabled
7-4 —	Reserved
3 HRDIR	Host Request Direction Configures the direction of the HREQ pin. The HREQ pin should only be configured as output in Slave mode. The HREQ pin direction is input for Master mode. 0b - HREQ pin is configured as input 1b - HREQ pin is configured as output
2 HRSEL	Host Request Select Selects the source of the host request input. When the host request function is enabled with the HREQ pin, the PCS[1] function is disabled. 0b - Host request input is the HREQ pin 1b - Host request input is the input trigger
1 HRPOL	Host Request Polarity Configures the polarity of the host request pin or trigger.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - HREQ pin or input trigger is active high 1b - HREQ pin or input trigger is active low
0 HREN	Host Request Enable When enabled in Master mode, LPSPI only starts a new SPI bus transfer if the host request input is asserted. When LPSPI is busy, the host request input is ignored. When enabled in Slave mode, the HREQ output pin asserts when data is available to be transmitted. 0b - Host request is disabled 1b - Host request is enabled

67.5.1.9 Configuration 1 (CFGR1)

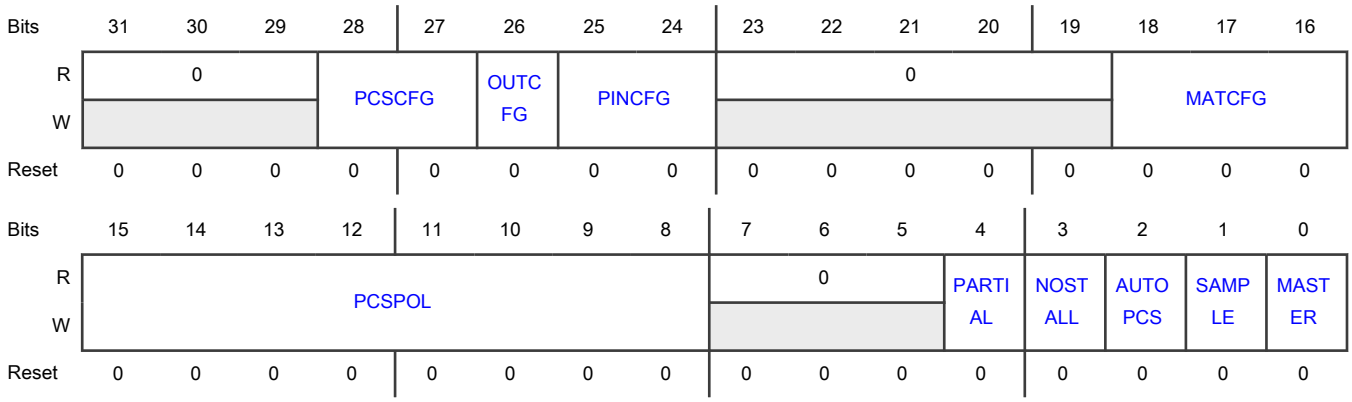
Offset

Register	Offset
CFGR1	24h

Function

CFGR1 should only be written when LPSPI is disabled.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-27	Peripheral Chip Select Configuration

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>PCSCFG</td><td>When performing parallel transfers, PCSCFG must be configured to enable the desired transfer.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>LPSPI_0</td><td>CFGR1</td><td>—</td></tr>
<tr><td>LPSPI_1</td><td>CFGR1[27]</td><td>CFGR1[28]</td></tr>
<tr><td>LPSPI_2</td><td>CFGR1[27]</td><td>CFGR1[28]</td></tr>
<tr><td>LPSPI_3</td><td>CFGR1[27]</td><td>CFGR1[28]</td></tr>
<tr><td>LPSPI_4</td><td>CFGR1[27]</td><td>CFGR1[28]</td></tr>
<tr><td>LPSPI_5</td><td>CFGR1[27]</td><td>CFGR1[28]</td></tr>
</table>

NOTE
The descriptions of the field settings vary by module instance.

<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>LPSPI_0</td><td>00b - PCS[7:2] are configured for chip select function
01b - PCS[3:2] are configured for half-duplex 4-bit transfers (PCS[3:2] = DATA[3:2])
11b - PCS[7:2] are configured for half-duplex 4-bit and 8-bit transfers (PCS[7:2] = DATA[7:2])</td></tr>
<tr><td>LPSPI_1</td><td>0b - PCS[5:2] are configured for chip select function
1b - PCS[3:2] are configured for half-duplex 4-bit transfers (PCS[3:2] = DATA[3:2])</td></tr>
<tr><td>LPSPI_2</td><td>0b - PCS[3:2] are configured for chip select function
1b - PCS[3:2] are configured for half-duplex 4-bit transfers (PCS[3:2] = DATA[3:2])</td></tr>
<tr><td>LPSPI_3</td><td>0b - PCS[3:2] are configured for chip select function
1b - PCS[3:2] are configured for half-duplex 4-bit transfers (PCS[3:2] = DATA[3:2])</td></tr>
<tr><td>LPSPI_4</td><td>0b - PCS[3:2] are configured for chip select function</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td></td><td>1b - PCS[3:2] are configured for half-duplex 4-bit transfers (PCS[3:2] = DATA[3:2])</td></tr>
<tr><td>LPSPI_5</td><td>0b - PCS[3:2] are configured for chip select function
1b - PCS[3:2] are configured for half-duplex 4-bit transfers (PCS[3:2] = DATA[3:2])</td></tr>
</table>
</td></tr>
<tr><td>26
OUTCFG</td><td>Output Configuration
Configures if the output data is tristated between accesses (PCS is negated). If performing half-duplex transfers, the Output Configuration bit must be set.
0b - Output data retains last value when chip select is negated
1b - Output data is tristated when chip select is negated</td></tr>
<tr><td>25-24
PINCFG</td><td>Pin Configuration
Configures which pins are used for input and output data during serial transfers. When performing parallel transfers, the Pin Configuration field is ignored.
00b - SIN is used for input data and SOUT is used for output data
01b - SIN is used for both input and output data, only half-duplex serial transfers are supported
10b - SOUT is used for both input and output data, only half-duplex serial transfers are supported
11b - SOUT is used for input data and SIN is used for output data</td></tr>
<tr><td>23-19
—</td><td>Reserved</td></tr>
<tr><td>18-16
MATCFG</td><td>Match Configuration
Configures the condition that causes the DMF to set.
NOTE
Syntax: * is boolean AND, + is boolean OR
000b - Match is disabled
001b - Reserved
010b - Match is enabled is 1st data word is MATCH0 or MATCH1. Match is enabled, if 1st data word equals MATCH0 OR MATCH1, that is (1st data word = MATCH0 + MATCH1)
011b - Match is enabled on any data word equal MATCH0 or MATCH1. Match is enabled, if any data word equals MATCH0 OR MATCH1, that is (any data word = MATCH0 + MATCH1)</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>100b - Match is enabled on data match sequence. Match is enabled, if 1st data word equals MATCH0 AND 2nd data word equals MATCH1, that is [(1st data word = MATCH0) * (2nd data word = MATCH1)]

101b - Match is enabled on data match sequence. Match is enabled, if any data word equals MATCH0 AND the next data word equals MATCH1, that is [(any data word = MATCH0) * (next data word = MATCH1)]

110b - Match is enabled. Match is enabled, if (1st data word AND MATCH1) equals (MATCH0 AND MATCH1), that is [(1st data word * MATCH1) = (MATCH0 * MATCH1)]

111b - Match is enabled. Match is enabled, if (any data word AND MATCH1) equals (MATCH0 AND MATCH1), that is [(any data word * MATCH1) = (MATCH0 * MATCH1)]</td></tr>
<tr><td>15-8
PCSPOL</td><td>Peripheral Chip Select Polarity

Configures the polarity of each Peripheral Chip Select pin. Each PCSPOL bit position (let's call it n) corresponds to a PCS[n] pin. For example, PCSPOL[0] is chip select 0 (at PCS[0] pin), and PCSPOL[1] is chip select 1 (at PCS[1] pin). If a PCSPOL bit:
• = 0, then the PCS[n] pin is active low.
• = 1, then the PCS[n] pin is active high.

NOTE
The entire PCSPOL field is not fully supported in every LPSPI module instance. Refer to the chip-specific information for LPSPI.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>LPSPI_0</td><td>CFGR1</td><td>—</td></tr>
<tr><td>LPSPI_1</td><td>CFGR1[13–8]</td><td>CFGR1[15–14]</td></tr>
<tr><td>LPSPI_2</td><td>CFGR1[11–8]</td><td>CFGR1[15–12]</td></tr>
<tr><td>LPSPI_3</td><td>CFGR1[11–8]</td><td>CFGR1[15–12]</td></tr>
<tr><td>LPSPI_4</td><td>CFGR1[11–8]</td><td>CFGR1[15–12]</td></tr>
<tr><td>LPSPI_5</td><td>CFGR1[11–8]</td><td>CFGR1[15–12]</td></tr>
</table>
</td></tr>
<tr><td>7-5
—</td><td>Reserved</td></tr>
<tr><td>4</td><td>Partial Enable</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
PARTIAL	In Slave mode, when PCS negates then any partially received word can be either discarded or stored in the receive FIFO. 0b - Partial words in the receive shift register when PCS negates are discarded. 1b - Partial words in the receive shift register when PCS negates are stored in the receive FIFO.
3 NOSTALL	No Stall In Master mode, LPSPI stalls transfers when the transmit FIFO is empty or when the receive FIFO is full, ensuring that no transmit FIFO underrun or receive FIFO overrun can occur. Setting the No Stall bit disables this functionality. 0b - Transfers stall when the transmit FIFO is empty or the receive FIFO is full 1b - Transfers do not stall, allowing transmit FIFO underruns or receive FIFO overruns to occur
2 AUTOPCS	Automatic PCS For correct operations, the LPSPI slave normally requires the PCS to negate between frames. Setting the AUTOPCS bit causes LPSPI to generate an internal PCS signal at the end of each transfer word when the Clock Phase bit TCR[CPHA] = 1. • When the Automatic PCS bit is set, the SCK must remain idle for at least 4 LPSPI functional clock cycles (divided by the Prescaler Value TCR[PRESCALE] configuration) between each word, to ensure correct operations • In Master mode, the Automatic PCS bit is ignored 0b - Automatic PCS generation is disabled 1b - Automatic PCS generation is enabled
1 SAMPLE	Sample Point When set, the LPSPI master samples the input data on a delayed loopback SCK clock edge, which improves the setup time when sampling data. • The input data setup time in Master mode with delayed SCK edge is equal to the input data setup time in Slave mode • In Slave mode, the SAMPLE bit is ignored 0b - Input data is sampled on SCK edge 1b - Input data is sampled on delayed SCK edge
0 MASTER	Master Mode Configures LPSPI in Master or Slave mode. The Master Mode bit directly controls the direction of the SCK and PCS pins. 0b - Slave mode 1b - Master mode

67.5.1.10 Data Match 0 (DMR0)

Offset

Register	Offset
DMR0	30h

Function

For the Data Match register to be available, set CFGR0[RDMO] = 1. Do not change the value stored in DMR0 while CFGR0[RDMO] is enabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MATCH0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MATCH0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MATCH0	Match 0 Value When CFGR0[RDMO] = 1, the value in MATCH0 is compared against the received data.

67.5.1.11 Data Match 1 (DMR1)

Offset

Register	Offset
DMR1	34h

Function

The DMR1 register use depends on setting CFGR0[RDMO] = 1. Do not change the value stored in DMR1 while CFGR0[RDMO] is enabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MATCH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MATCH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MATCH1	Match 1 Value When CFGR0[RDMO] = 1, the value in MATCH1 is compared against the received data.

67.5.1.12 Clock Configuration (CCR)

Offset

Register	Offset
CCR	40h

Function

The Clock Configuration Register is only used in Master mode, and the Clock Configuration Register cannot be changed when LPSPI is enabled.

Diagram

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

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

Fields

Field	Function
31-24 SCKPCS	SCK-to-PCS Delay In Master mode: configures the delay from the last SCK edge to the PCS negation. • The delay is equal to (SCKPCS + 1) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The minimum delay is 1 cycle. See Figure 362.
23-16 PCSSCK	PCS-to-SCK Delay In Master mode: configures the delay from the PCS assertion to the first SCK edge. • The delay is equal to (PCSSCK + 1) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The minimum delay is 1 cycle. See Figure 362.
15-8 DBT	Delay Between Transfers Writing this field updates the contents of CCR1[PCSPCS] and CCR1[SCKSCK].
7-0 SCKDIV	SCK Divider Writing this field updates the contents of CCR1[SCKSET] and CCR1[SCKHLD].

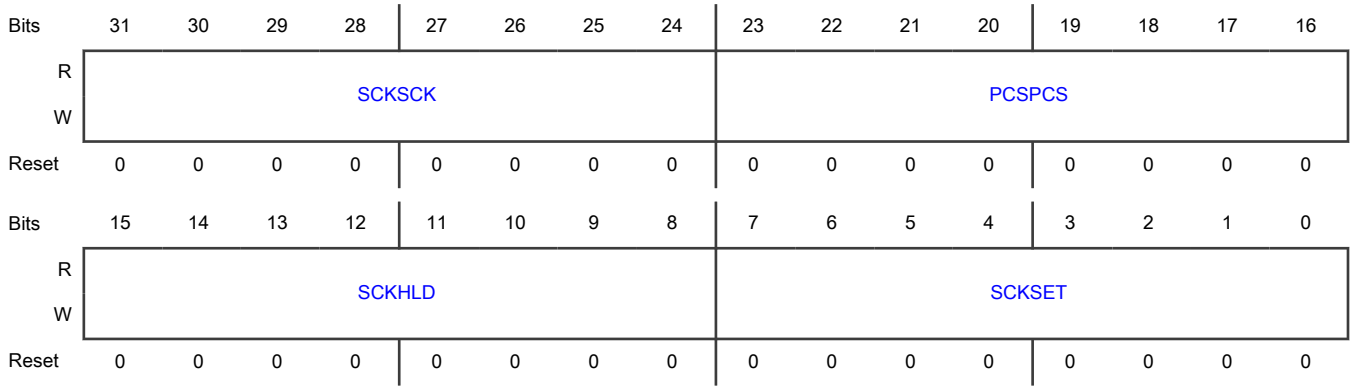
67.5.1.13 Clock Configuration 1 (CCR1)

Offset

Register	Offset
CCR1	44h

Function
The Clock Configuration Register 1 is only used in Master mode, and the Clock Configuration Register cannot be changed when LPSPI is enabled.

Diagram



Fields

Field	Function
31-24 SCKSCK	SCK Inter-Frame Delay In Master mode: • Configures the delay from the last SCK pulse of a frame and the first SCK pulse of the following frame, in a continuous transfer. • The delay is equal to (SCKSCK + 1) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The minimum delay is 1 cycle. **NOTE** For backwards compatibility, writing CCR[DBT] updates CCR1[SCKSCK] with the value written.
23-16 PCSPCS	PCS to PCS delay In Master mode: • Configures the delay from the PCS negation to the next PCS assertion. • The delay is equal to (PCSPCS + PCSPCS + 2) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The minimum delay is 2 cycles. • Half of the delay (PCSPCS + 1) occurs before PCS assertion and the other half of the delay (PCSPCS + 1) occurs after PCS negation. If the command word is updated between 2 transfers, then the command word is updated half-way between the PCS negation of the last transfer and PCS assertion of the next transfer. • The command word sets which PCS signal is used, the polarity/phase of the SCK signal, and the Prescaler Value. **NOTE** For backwards compatibility, writing CCR[DBT] updates CCR1[PCSPCS] with (DBT/2) rounded up.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 SCKHLD	SCK Hold In Master mode, the SCK Hold configures the hold phase of the SCK pin. • The hold phase is the delay between the SCK edge that samples the receive data and the SCK edge that drives the transmit data. • This is the SCK low period for CPHA=0, CPOL=1 and CPHA=1, CPOL =0 and the SCK high period for CPHA=0, CPOL=0 and CPHA=1, CPOL=1. • The SCK hold phase delay is equal to (SCKHLD + 1) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The minimum delay is 1 cycle. • The SCK period is equal to (SCKSET + SCKHLD + 2) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The SCK duty cycle is based on the difference between SCKSET and SCKHLD, configure both fields to the same value for 50/50 duty cycle. See Figure 362. **NOTE** For backwards compatibility, writing CCR[SCKDIV] updates CCR1[SCKHLD] with (SCKDIV/2) rounded down.
7-0 SCKSET	SCK Setup In Master mode, the SCK Setup configures the setup phase of the SCK pin. • The setup phase is the delay between the SCK edge that drives the transmit data and the SCK edge that samples the receive data. • This is the SCK high period for CPHA=0, CPOL=1 and CPHA=1, CPOL =0 and the SCK low period for CPHA=0, CPOL=0 and CPHA=1, CPOL=1. • The SCK setup phase delay is equal to (SCKSET + 1) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The minimum delay is 1 cycle. • The SCK period is equal to (SCKSET + SCKHLD + 2) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The SCK duty cycle is based on the difference between SCKSET and SCKHLD, configure both fields to the same value for 50/50 duty cycle. See Figure 362. **NOTE** For backwards compatibility, writing CCR[SCKDIV] updates CCR1[SCKSET] with (SCKDIV/2) rounded up.

67.5.1.14 FIFO Control (FCR)

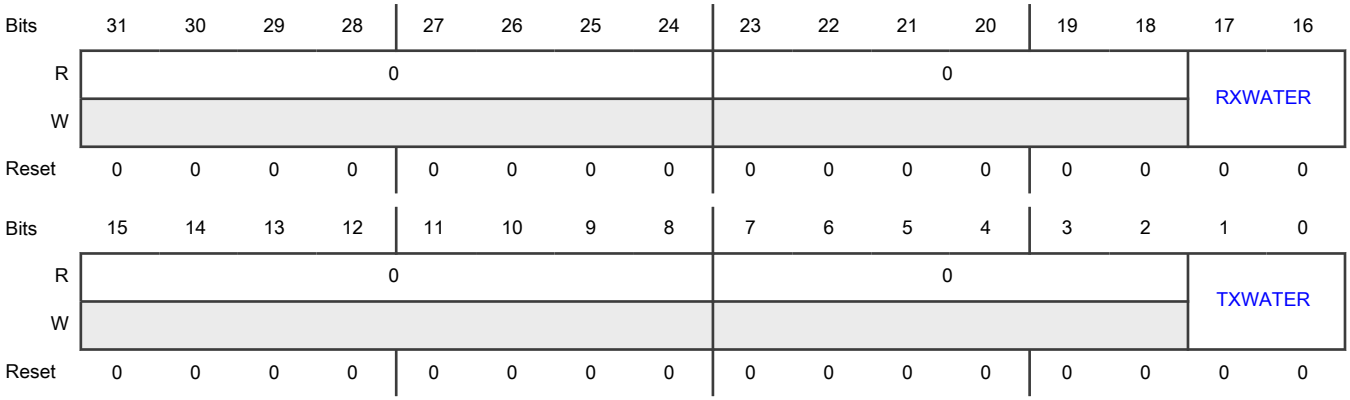
Offset

Register	Offset
FCR	58h

Function

Contains the RXWATER and TXWATER control fields.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-18 —	Reserved
17-16 RXWATER	Receive FIFO Watermark The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. Writing a value equal or greater than the FIFO size truncates the written value.
15-8 —	Reserved
7-2 —	Reserved
1-0 TXWATER	Transmit FIFO Watermark The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. Writing a value equal or greater than the FIFO size truncates the written value.

67.5.1.15 FIFO Status (FSR)

Offset

Register	Offset
FSR	5Ch

Function

Contains the TXCOUNT and RXCOUNT fields, which hold the number of words currently stored in the FIFO.

Diagram

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

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

Fields

Field	Function
31-24 —	Reserved
23-19 —	Reserved
18-16 RXCOUNT	Receive FIFO Count Returns the number of words currently stored in the receive FIFO.
15-8 —	Reserved
7-3 —	Reserved
2-0 TXCOUNT	Transmit FIFO Count Returns the number of words currently stored in the transmit FIFO.

67.5.1.16 Transmit Command (TCR)

Offset

Register	Offset
TCR	60h

Function

Writes to either the Transmit Command Register or Transmit Data Register pushes the data into the transmit FIFO, in the order that the data are written. The Command Register should only be written using 32-bit writes. Command Register writes are tagged and cause the command register to update, after that entry reaches the top of the FIFO. This allows changes to the command word and the transmit data itself to be interleaved. That is, the writes to the two registers can be interleaved (write command word, then data word, then command word, and so on). Changing the command word causes all subsequent SPI bus transfers to be performed using the new command word.

- **In Master mode,** writing a new command word does not initiate a new transfer, unless TXMSK is set. Transfers are initiated by transmit data in the transmit FIFO, or by a new command word (with TXMSK set). Hardware clears TXMSK when the PCS negates.
- **In Master mode,** if the command word is changed before an existing frame has completed, then the existing frame terminates and the command word then updates. The command word can be changed during a continuous transfer, if CONTC of the new command word is set and the command word is written on a frame size boundary.
- **In Slave mode,** the command word should be changed only when LPSPI is idle and there is no SPI bus transfer.

Avoid register reading problems: Reading the Transmit Command Register returns the current state of the command register. Reading the Transmit Command Register at the same time that the Transmit Command Register is loaded from the transmit FIFO, can return an incorrect Transmit Command Register value. It is recommended:

- Read the Transmit Command Register when the transmit FIFO is empty,
- Read the Transmit Command Register more than once and then compare the returned values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CPOL	CPHA	PRESCALE			PCS			LSBF	BYSW	CONT	CONTC	RXMSK	TXMSK	WIDTH	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31	Clock Polarity

Table continues on the next page...

Table continued from the previous page...

Field	Function
CPOL	The Clock Polarity field is only updated when PCS negated. See Figure 362. 0b - The inactive state value of SCK is low 1b - The inactive state value of SCK is high
30 CPHA	Clock Phase The Clock Phase field is only updated when PCS negated. See Figure 362. 0b - Captured. Data is captured on the leading edge of SCK and **changed** on the following edge of SCK 1b - Changed. Data is changed on the leading edge of SCK and **captured** on the following edge of SCK
29-27 PRESCALE	Prescaler Value For all SPI bus transfers, the Prescaler value applied to the clock configuration register. The Prescaler Value field is only updated when PCS negated. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128
26-24 PCS	Peripheral Chip Select Configures the peripheral chip select used for the transfer. The Peripheral Chip Select field is only updated when PCS negated. **NOTE** The entire PCS field is not fully supported in every LPSPI module instance. See the chip-specific LPSPI information. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
LPSPI_0	TCR	—

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
LPSPI_1	TCR	—
LPSPI_2	TCR[25–24]	TCR[26]
LPSPI_3	TCR[25–24]	TCR[26]
LPSPI_4	TCR[25–24]	TCR[26]
LPSPI_5	TCR[25–24]	TCR[26]

NOTE

The descriptions of the field settings vary by module instance.

Instance	Field value and description
LPSPI_0	000b - Transfer using PCS[0] 001b - Transfer using PCS[1] 010b - Transfer using PCS[2] 011b - Transfer using PCS[3] 100b - Transfer using PCS[4] 101b - Transfer using PCS[5] 110b - Transfer using PCS[6] 111b - Transfer using PCS[7]
LPSPI_1	000b - Transfer using PCS[0] 001b - Transfer using PCS[1] 010b - Transfer using PCS[2] 011b - Transfer using PCS[3] 100b - Transfer using PCS[4] 101b - Transfer using PCS[5] 110b - Transfer using PCS[6] 111b - Transfer using PCS[7]
LPSPI_2	00b - Transfer using PCS[0] 01b - Transfer using PCS[1] 10b - Transfer using PCS[2]

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td></td><td>11b - Transfer using PCS[3]</td></tr>
<tr><td>LPSPI_3</td><td>00b - Transfer using PCS[0]
01b - Transfer using PCS[1]
10b - Transfer using PCS[2]
11b - Transfer using PCS[3]</td></tr>
<tr><td>LPSPI_4</td><td>00b - Transfer using PCS[0]
01b - Transfer using PCS[1]
10b - Transfer using PCS[2]
11b - Transfer using PCS[3]</td></tr>
<tr><td>LPSPI_5</td><td>00b - Transfer using PCS[0]
01b - Transfer using PCS[1]
10b - Transfer using PCS[2]
11b - Transfer using PCS[3]</td></tr>
</table>
</td></tr>
<tr><td>23
LSBF</td><td>LSB First
0b - Data is transferred MSB first
1b - Data is transferred LSB first</td></tr>
<tr><td>22
BYSW</td><td>Byte Swap
Byte swap swaps the contents of [31:24] with [7:0] and [23:16] with [15:8] for each transmit data word read from the FIFO and for each received data word stored to the FIFO (or compared with match registers).
0b - Byte swap is disabled
1b - Byte swap is enabled</td></tr>
<tr><td>21
CONT</td><td>Continuous Transfer
• In Master mode, CONT keeps the PCS asserted at the end of the frame size, until a command word is received that starts a new frame.
• In Slave mode, when CONT is enabled, LPSPI only transmits the first FRAMESZ bits; after which LPSPI transmits received data (assuming a 32-bit shift register) until the next PCS negation.
0b - Continuous transfer is disabled
1b - Continuous transfer is enabled</td></tr>
<tr><td>20</td><td>Continuing Command</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>CONTC</td><td>In Master mode, the CONTC bit allows the command word to be changed within a continuous transfer.
The initial command word must enable continuous transfer (CONT = 1),
the continuing command must set this bit (CONTC = 1),
and the continuing command word must be loaded on a frame size boundary.
For example, if the continuous transfer has a frame size of 64-bits, then a continuing command word must be loaded on a 64-bit boundary.
In Slave mode, the Continuing Command bit modifies the internal RXMSK/TXMSK configuration after the first FRAMESZ bits and until the PCS negation.
Receive data is discarded after the first FRAMESZ bits, if CONT is also set this does not block the transmission of received data.
Transmit data is not masked after the first FRAMESZ bits, this allows the first FRAMESZ bits to be received and a response transmitted.
0b - Command word for start of new transfer
1b - Command word for continuing transfer</td></tr>
<tr><td>19
RXMSK</td><td>Receive Data Mask
When set, receive data is masked (receive data is not stored in receive FIFO).
0b - Normal transfer
1b - Receive data is masked</td></tr>
<tr><td>18
TXMSK</td><td>Transmit Data Mask
When set, transmit data is masked (no data is loaded from transmit FIFO and output pin is tristated). In Master mode, the TXMSK bit initiates a new transfer which cannot be aborted by another command word; the TXMSK bit is cleared by hardware at the end of the transfer.
0b - Normal transfer
1b - Mask transmit data</td></tr>
<tr><td>17-16
WIDTH</td><td>Transfer Width
Configures between serial (1-bit) or parallel transfers. For half-duplex parallel transfers, either Receive Data Mask (RXMSK) or Transmit Data Mask (TXMSK) must be set.
NOTE
The descriptions of the field settings vary by module instance.<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>LPSPI_0</td><td>00b - 1 bit transfer
01b - 2 bit transfer
10b - 4 bit transfer</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function	
	Instance	Field value and description
		11b - 8 bit transfer
	LPSPI_1	00b - 1 bit transfer 01b - 2 bit transfer 10b - 4 bit transfer 11b - Reserved
	LPSPI_2	00b - 1 bit transfer 01b - 2 bit transfer 10b - 4 bit transfer 11b - Reserved
	LPSPI_3	00b - 1 bit transfer 01b - 2 bit transfer 10b - 4 bit transfer 11b - Reserved
	LPSPI_4	00b - 1 bit transfer 01b - 2 bit transfer 10b - 4 bit transfer 11b - Reserved
	LPSPI_5	00b - 1 bit transfer 01b - 2 bit transfer 10b - 4 bit transfer 11b - Reserved
15-12 —	Reserved	
11-0 FRAMESZ	Frame Size Configures the frame size in number of bits equal to (FRAMESZ + 1). • The minimum frame size is 8 bits, or 16 bits for an 8-bit transfer • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32-bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately.	

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contains the remainder bits. For example, a 72-bit transfer consists of 3 words: the 1st and 2nd words are 32 bits, and the 3rd word is 8 bits.

67.5.1.17 Transmit Data (TDR)

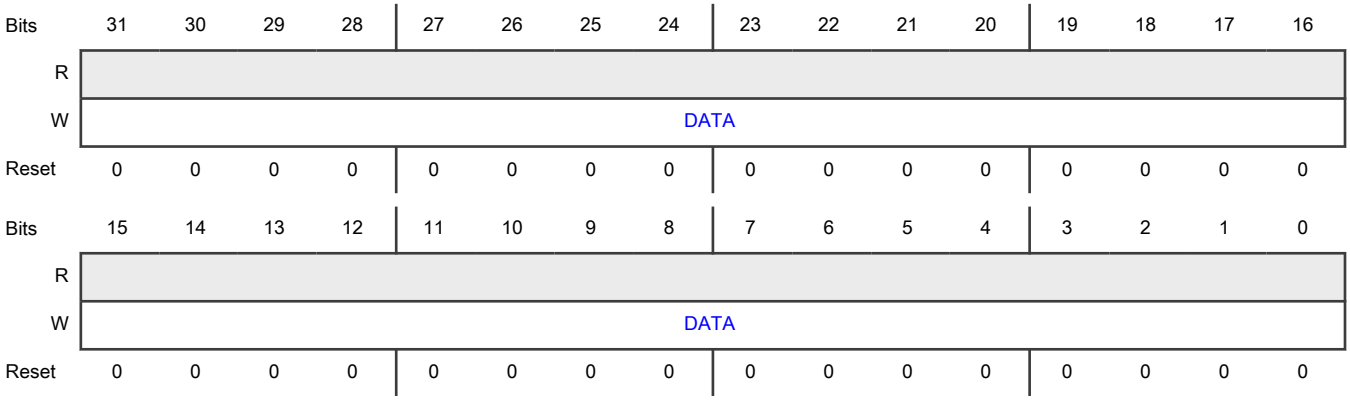
Offset

Register	Offset
TDR	64h

Function

Writes to either the Transmit Command Register or Transmit Data Register push the data into the transmit FIFO, in the order that the data was written. The Data Register can be written using 32-bit, 16-bit, or 8-bit writes, but each write pushes data into the FIFO with zero pushed in unwritten bytes.

Diagram



Fields

Field	Function
31-0 DATA	Transmit Data Both 8-bit and 16-bit writes of transmit data zero-extend the data written and push the data into the transmit FIFO. To zero-extend 8-bit and 16-bit writes (to 32 bits), means that the higher order (most significant) empty parts of the 8-bit and 16-bit writes are filled with zeroes.

67.5.1.18 Receive Status (RSR)

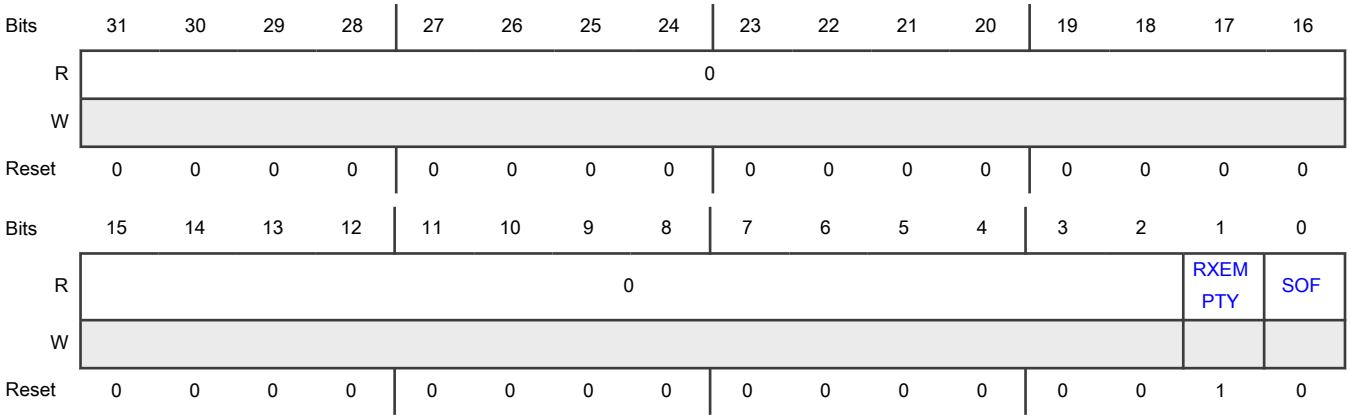
Offset

Register	Offset
RSR	70h

Function

Contains data flow status fields for receive FIFO.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RXEMPTY	RX FIFO Empty 0b - RX FIFO is not empty 1b - RX FIFO is empty
0 SOF	Start Of Frame Indicates that this is the first data word received after PCS assertion. 0b - Subsequent data word received after PCS assertion 1b - First data word received after PCS assertion

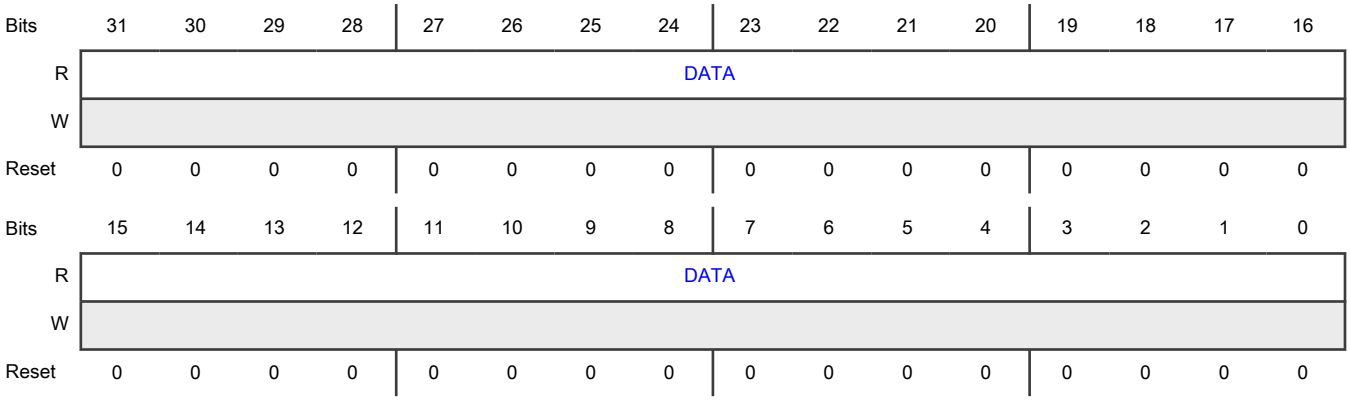
67.5.1.19 Receive Data (RDR)

Offset

Register	Offset
RDR	74h

Function
Pulls the first entry from the receive FIFO.

Diagram



Fields

Field	Function
31-0 DATA	Receive Data

67.5.1.20 Receive Data Read Only (RDROR)

Offset

Register	Offset
RDROR	78h

Function
Returns the first entry in the receive FIFO, but does not pull the data from the FIFO.

Diagram

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

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

Fields

Field	Function
31-0 DATA	Receive Data

67.5.1.21 Transmit Command Burst (TCBR)

Offset

Register	Offset
TCBR	3FCh

Function

The Transmit Command Burst Register is an alias of the Transmit Command Register designed to support burst transfers to the transmit FIFO for use with the DMA controller.

Diagram

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

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

Fields

Field	Function
31-0 DATA	Command Data Data is written to the Transmit Command Register.

67.5.1.22 Transmit Data Burst (TDBR0 - TDBR127)

Offset

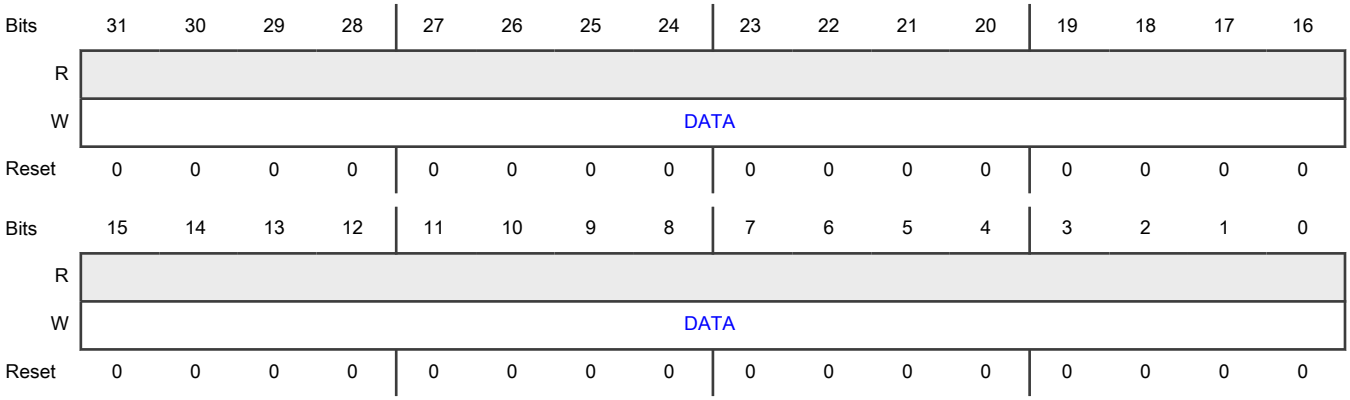
For n = 0 to 127:

Register	Offset
TDBRn	400h + (n × 4h)

Function

The Transmit Data Burst Register is an alias of the Transmit Data Register designed to support burst transfers to the transmit FIFO for use with the DMA controller. The size of the Transmit Data Burst Register is 512-bytes.

Diagram



Fields

Field	Function
31-0 DATA	Data Data is written to the Transmit Data Register.

67.5.1.23 Receive Data Burst (RDBR0 - RDBR127)

Offset

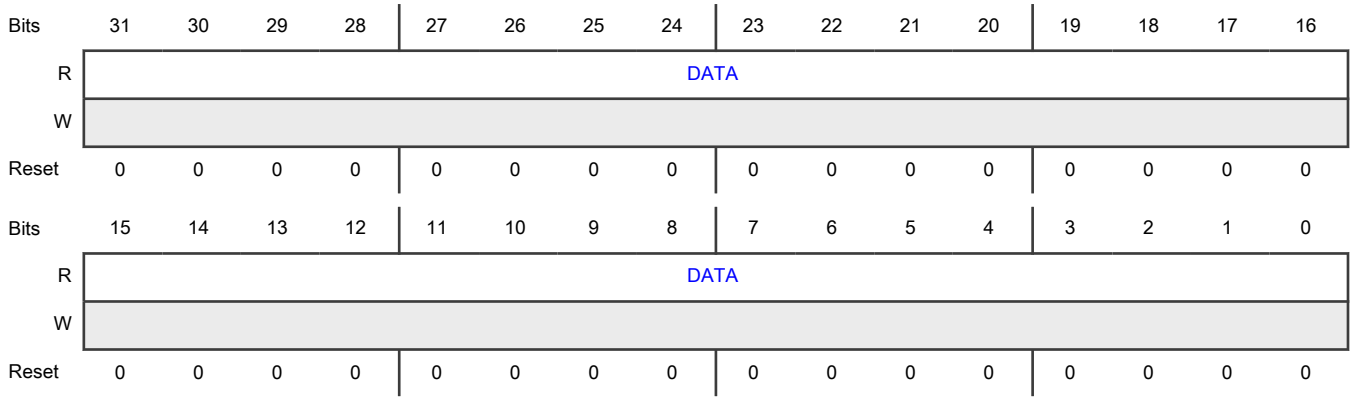
For n = 0 to 127:

Register	Offset
RDBRn	600h + (n × 4h)

Function

The Receive Data Burst Register is an alias of the Receive Data Register designed to support burst transfers from the receive FIFO. The size of the Receive Data Burst Register is 512-bytes.

Diagram



Fields

Field	Function
31-0 DATA	Data Data is read from the Receive Data Register.

67.6 Glossary

PCS	Peripheral chip select
SCK	Serial clock
SDI	Slave data in
SDO	Slave data out
SS	Slave select

Chapter 68
Low Power Inter-Integrated Circuit (LPI2C)

68.1 Chip-specific LPI2C information

68.1.1 LPI2C instances and configuration

This chip has two instances of LPI2C: LPI2C_0 and LPI2C_1.

Table 376. LPI2C configuration

Instances	TX FIFO Size	RX FIFO Size	SMBus[1]	Slave Mode Enable
LPI2C_0	4x11 bit	4x8 bit	Yes	Yes
LPI2C_1	4x11 bit	4x8 bit	Yes	Yes

1. System Management Bus

- LPI2C slave mode operation supports up to high speed mode = 3.4MHz (3.4 Mbps data rate; effective data rate reduces according to I2C protocol).
- LPI2C master mode operation supports up to fast Mode = 400KHz (400 kbps data rate; effective data rate reduces according to I2C protocol).
- The LPI2C module includes SMBus (System Management Bus) support and DMA support.
- LPI2C RX FIFO is 8-bit and TX FIFO is 11-bit (8bit data + 3bit command)
- LPI2C supports fast data communication as per I2C standard v3.0.

68.2 Overview

The LPI2C is a low power Inter-Integrated Circuit (I2C) module that supports an efficient interface to an I^2C bus as a master and/or as a slave.

- Implements logic support for standard-mode, fast-mode, fast-mode plus and ultra-fast modes of operation.
- Uses little CPU overhead, with DMA offloading of FIFO register accesses.

The LPI2C module also complies with the System Management Bus (SMBus) Specification, version 3. The SMBus is a single-ended simple two-wire bus, which is typically used for low bandwidth communications.

NOTE

The I^2C (Inter-Integrated Circuit) serial bus is multi-master, multi-slave, packet-switched, and single-ended, and is often used to attach microcontroller ICs to lower-speed peripheral ICs.

68.2.1 Block diagram

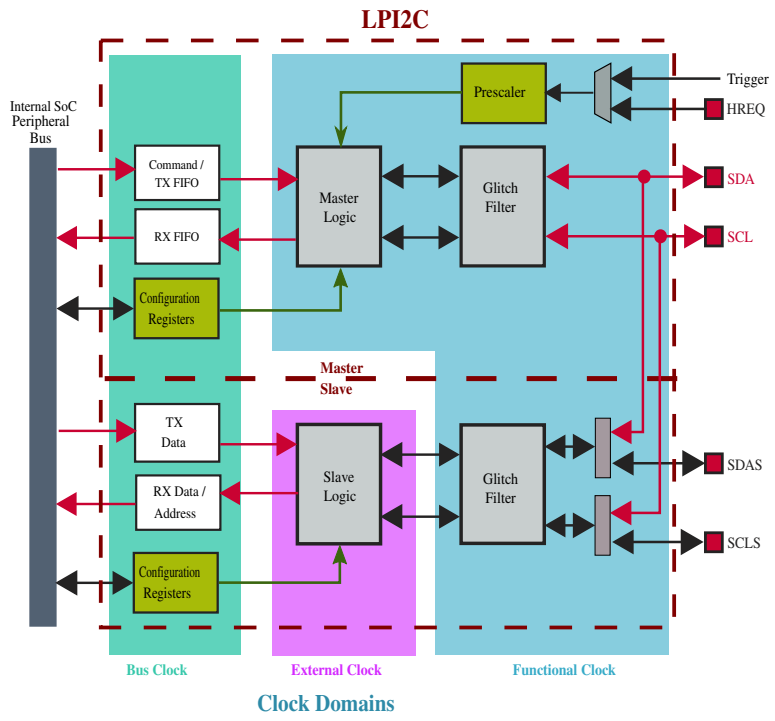


Figure 364. LPI2C block diagram

68.2.2 Features

The LPI2C supports:

- Standard, Fast, Fast+ and Ultra Fast modes are supported
- High speed mode (HS) in slave mode
- Multi-master support, including synchronization and arbitration. Multi-master means any number of master nodes can be present. Additionally, master and slave roles may be changed between messages (after a STOP is sent).
- Clock stretching: Sometimes multiple I2C nodes may be driving the lines at the same time. If any I2C node is driving a line low, then that line will be low. I2C nodes that are starting to transmit a logical one (by letting the line float high) can detect that the line is low, and thereby know that another I2C node is active at the same time.
  - When node detection is used on the SCL line, it is called *clock stretching*, and clock stretching is used as a I2C flow control mechanism.
  - When node detection is used on the SDA line, it is called *arbitration*, and arbitration ensures that there is only one I2C node transmitter at a time.
- General call, 7-bit and 10-bit addressing
- Software reset, START byte and Device ID (also require software support)

The LPI2C master supports:

- Command/transmit FIFO of 4 words (8-bit transmit data + 3-bit command).
- Receive FIFO of 4 words (8-bit receive data).
- Command FIFO will wait for idle I2C bus before initiating transfer
- Command FIFO can initiate (repeated) START and STOP conditions and one or more master-receiver transfers

- STOP condition can be generated from command FIFO, or generated automatically when the transmit FIFO is empty
- Host request input to control the start time of an I2C bus transfer
- Flexible receive data match can generate interrupt on data match and/or discard unwanted data
- Flag and optional interrupt to signal Repeated START condition, STOP condition, loss of arbitration, unexpected NACK, and command word errors
- Supports configurable bus idle timeout and pin-stuck-low timeout

The LPI2C slave supports:

- Separate I2C slave registers to minimize software overhead because of master/slave switching
- Support for 7-bit or 10-bit addressing, address range, SMBus alert and general call address
- Transmit data register that supports interrupt or DMA requests
- Receive data register that supports interrupt or DMA requests
- Software-controllable ACK or NACK, with optional clock stretching on ACK/NACK bit
- Configurable clock stretching, to avoid transmit FIFO underrun and receive FIFO overrun errors
- Flag and optional interrupt at end of packet, STOP condition, or bit error detection

68.3 Functional description

68.3.1 Master Mode

The LPI2C master logic operates independently from the slave logic to perform all master mode transfers on the I2C bus.

Transmit and Command FIFO commands

The transmit FIFO stores command data to initiate the various I2C operations. The following operations can be initiated through commands in the transmit FIFO:

- START or Repeated START condition with address byte and expecting ACK or NACK.
- Transmit data (this is the default for zero extended byte writes to the transmit FIFO).
- Receive 1-256 bytes of data (can also be configured to discard receive data and not store in receive FIFO).
- STOP condition (can also be configured to send STOP condition when transmit FIFO is empty).

Multiple transmit and receive commands can be inserted between the START condition and STOP conditon; transmit and receive commands must not be interleaved (to comply with the I2C specification). The receive data command and the receive data and discard commands can be interleaved, to ensure that only the desired received data is stored in the receive FIFO (or compared with the data match logic).

The LPI2C master automatically transmits a NACK on the last byte of a receive data command unless the next command in the FIFO is also a receive data command. A NACK is also automatically transmitted if the transmit FIFO is empty when a receive data command completes.

The LPI2C master supports 10-bit addressing through a (repeated) START condition, followed by a transmit data byte containing the second address byte, followed by any number of data bytes with the master-transmit data.

A START or Repeated START condition that is expecting a NACK (for example,HS-mode master code) must be followed by a STOP or (repeated) START condition.

Master operations

Whenever the LPI2C is enabled, it monitors the I2C bus to detect when the I2C bus is idle (MSR[BBF]). The I2C bus is no longer considered idle if either SCL or SDA are low, and the I2C bus becomes idle if a STOP condition is detected or if a bus idle timeout

is detected (as configured by MCFGR2[BUSIDLE]). After the I2C bus is idle, the transmit FIFO is not empty, and the host request is either asserted or disabled, then the LPI2C master initiates a transfer on the I2C bus. This involves the following steps:

- Wait the bus idle time equal to (MCCR0[CLKLO] + 1) multiplied by the prescaler (MCFGR1[PRESCALE]).
- Transmit a START condition and address byte using the timing configuration in Master Clock Configuration 0 (MCCR0); if a high speed mode transfer is configured, then the timing configuration from Master Clock Configuration 1 (MCCR1) is used instead.
- Perform master-transmit or master-receive transfers, as configured by the transmit FIFO.
- Transmit NACK on the last byte of a master-receive transfer, unless the next command in the transmit FIFO is also a receive data command and the transmit FIFO is not empty.
- Transmit a Repeated START or STOP condition as configured by the transmit FIFO and/or MCFGR1[AUTOSTOP]. A repeated START can change which timing configuration register is used.

When the LPI2C master is disabled (either due to MCR[MEN] being clear or automatically due to mode entry), the LPI2C continues to empty the transmit FIFO until a STOP condition is transmitted. However, the LPI2C will no longer stall the I2C bus waiting for the transmit or receive FIFO, and after the transmit FIFO is empty, the LPI2C generates a STOP condition automatically.

The LPI2C master can stall the I2C bus under certain conditions; this results in SCL pulled low continuously on the first bit of a byte, until the condition is removed:

- LPI2C master is enabled and busy, the transmit FIFO is empty, and MCFGR1[AUTOSTOP] is clear.
- LPI2C master is enabled and receiving data, receive data is not being discarded (due to command or receive data match), and the receive FIFO is full.

Receive FIFO and Data Matching

The receive FIFO is used to store receive data during master-receiver transfers. Receive data can also be configured to discard receive data instead of storing in the receive FIFO; this is configured by the command word in the transmit FIFO.

Receive data supports a receive data match function that can match received data against one of two bytes or against a masked data byte. The data match function can also be configured to compare only the first one or two received data words since the last (repeated) START condition. Receive data that is already discarded due to the command word cannot cause the data match to set, and delaya the match on the first received data word until after the discarded data is received. The receiver match function can also be configured to discard all receive data until a data match is detected, using the MCFGR0[RDMO] control bit. When clearing the MCFGR0[RDMO] control bit following a data match, clear MCFGR0[RDMO] before clearing MSR[DMF], to allow all subsequent data to be received.

Timing Parameters

The following timing parameters can be configured by the LPI2C master. Parameters are configured separately for high speed mode (Master Clock Configuration 1 (MCCR1)) and other modes (Master Clock Configuration 0 (MCCR0)). This allows the high speed mode master code to be sent using the regular timing parameters, and then switch to the high speed mode timing (following a repeated START) until the next STOP condition.

The LPI2C master timing parameters in LPI2C functional clock cycles are configured as follows. They must be configured to meet the I2C timing specification for the required mode.

Table 377. Timing Parameters

I2C Specification Timing Parameter	I2C Specification Timing Symbol	LPI2C Timing Parameter (LPI2C functional clock cycles)
SCL clock period	tSCL	(CLKHI + CLKLO + 2 + SCL_LATENCY) x (2 ^ PRESCALE)
hold time (repeated) START condition	tHD:STA	(SETHOLD +1) x (2 ^ PRESCALE)

Table continues on the next page...

Table 377. Timing Parameters (continued)

I2C Specification Timing Parameter	I2C Specification Timing Symbol	LPI2C Timing Parameter (LPI2C functional clock cycles)
LOW period of the SCL clock	tLOW	(CLKLO + 1) x (2 ^ PRESCALE)
HIGH period of the SCL clock	tHIGH	(CLKHI + 1 + SCL_LATENCY) x (2 ^ PRESCALE)
setup time for a repeated START condition or STOP condition	tSU:STA, tSU:STO	(SETHOLD + 1 + SCL_LATENCY) x (2 ^ PRESCALE)
data hold time	tHD:DAT	(DATAVD + 1) x (2 ^ PRESCALE)
data setup time	tSU:DAT	(SDA_LATENCY + 1) x (2 ^ PRESCALE)
bus free time between a STOP and START condition	tBUF	(CLKLO + 1 + SDA_LATENCY) x (2 ^ PRESCALE)
data valid time, data valid acknowledge time	tVD:DAT, tVD:ACK	(DATAVD + 1) x (2 ^ PRESCALE)

The latency parameters are defined in the following table, these parameters assume the risetime is less than one LPI2C functional clock cycle. The risetime depends on a number of factors, including the I/O propagation delay, the I2C bus loading and the external pull-up resistor sizing. A larger rise time increasea the number of cycles that the signal takes to propagate through the synchronizer (and glitch filter), which increases the latency.

Table 378. Synchronization Latency

Timing Parameter	Timing Definition
SCL_LATENCY	ROUNDDOWN ((2 + FILTSCL + SCL_RISETIME) / (2 ^ PRESCALE))
SDA_LATENCY	ROUNDDOWN ((2 + FILTSDA + SDA_RISETIME) / (2 ^ PRESCALE))

The following timing restrictions must be enforced to avoid unexpected START or STOP conditions on the I2C bus, or to avoid unexpected START or STOP conditions detected by the LPI2C master. The timing restrictions can be summarized as **SDA cannot change when SCL is high outside of a transmitted (repeated) START or STOP condition**.

Table 379. LPI2C Timing Parameter Restrictions

Timing Parameter	Minimum	Maximum	Comment
CLKLO	0x03	-	Also: CLKLO x (2 ^ PRESCALE) > SCL_LATENCY
CLKHI	0x01	-	Configure CLKHI to meet the duty cycle requirements in the I2C specification
SETHOLD	0x02	-	Also: SETHOLD x (2 ^ PRESCALE) > SDA_LATENCY
DATAVD	0x01	CLKLO - SDA_LATENCY - 1	Configure DATAVD to meet the data hold requirement in the I2C specification
FILTSCL	0x00	[CLKLO x (2 ^ PRESCALE)] - 3	FILTSCL and FILTSDA are the only parameters not multiplied by (2 ^ PRESCALE)

Table continues on the next page...

Table 379. LPI2C Timing Parameter Restrictions (continued)

Timing Parameter	Minimum	Maximum	Comment
FILTSDA	FILTSCL	[CLKLO x (2 ^ PRESCALE)] - 3	Configuring FILTSDA greater than FILTSCL can delay the SDA input to compensate for board level skew
BUSIDLE	(CLKLO+SETHOLD+2) x 2	-	Must also be greater than (CLKHI+1)

The timing parameters must be configured to meet the requirements of the I2C specification; this depends on the mode being supported and the LPI2C functional clock frequency. When switching between two modes using the different clock configuration registers (for example, Fast and HS-mode), the PRESCALE factor must remain constant between the modes. Some example timing configurations are provided below.

Table 380. LPI2C Example Timing Configurations

I2C Mode	Clock Frequency	Baud Rate	PRESCALE	FILTSCL / FILTSDA	SETHOLD	CLKLO	CLKHI	DATAVD
Fast	8 MHz	400 kbps	0x0	0x0/0x0	0x04	0x0B	0x05	0x02
Fast+	8 MHz	1 Mbps	0x0	0x0/0x0	0x02	0x03	0x01	0x01
Fast	48 MHz	400 kbps	0x0	0x1/0x1	0x1D	0x3E	0x35	0x0F
Fast	48 MHz	400 kbps	0x2	0x1/0x1	0x07	0x11	0x0B	0x03
Fast+	48 MHz	1 Mbps	0x2	0x1/0x1	0x03	0x06	0x04	0x04
HS-mode	48 MHz	3.2 Mbps	0x0	0x0/0x0	0x07	0x08	0x03	0x01
Fast	60 MHz	400 kbps	0x1	0x2/0x2	0x11	0x28	0x1F	0x08
Fast+	60 MHz	1 Mbps	0x1	0x2/0x2	0x07	0x0F	0x0B	0x01
HS-mode	60 MHz	3.33 Mbps	0x1	0x0/0x0	0x04	0x04	0x02	0x01

Error Conditions

The LPI2C master monitors for errors while it is active, the following conditions generates an error flag and block a new START condition from being sent, until the flag is cleared by software:

- A START or STOP condition is detected and is not generated by the LPI2C master (sets MSR[ALF]).
- Transmitting data on SDA and different values are being received (sets MSR[ALF]).
- NACK is detected when transmitting data, and MCFGR1[IGNACK] is clear (sets MSR[NDF]).
- NACK is detected and is expecting ACK for the address byte, and MCFGR1[IGNACK] is clear (sets MSR[NDF]).
- ACK is detected and is expecting NACK for the address byte, and MCFGR1[IGNACK] is clear (sets MSR[NDF]).
- Transmit FIFO is requesting to transmit or receive data without a START condition (sets MSR[FEF]).
- SCL (or SDA if MCFGR1[TIMECFG] is set) is low for (MCFGR2[TIMELOW] * 256) prescaler cycles without a pin transition (sets MSR[PLTF]).

Software must respond to the MSR[PLTF] flag to terminate the existing command either cleanly (by clearing MCR[MEN]), or abruptly (by setting MCR[RST]).

The MCFGR2[BUSIDLE] field can be used to force the I2C bus to be considered idle when SCL and SDA remain high for (BUSIDLE+1) prescaler cycles. The I2C bus is normally considered idle when the LPI2C master is first enabled, but when

BUSIDLE is configured greater than zero then SCL and/or SDA must be high for (BUSIDLE+1) prescaler cycles before the I2C bus is first considered idle.

Pin Configuration

- **Open-drain support:** The LPI2C master defaults to open-drain configuration of the SDA and SCL pins. Support for true open drain depends on the specific device, and requires the pins where LPI2C pins are muxed to support true open drain.
- **High Speed mode support:** Support for high speed mode also depends on the specific device, and requires the SCL pin to support the current source pull-up required in the I2C specification.
- **Ultra-Fast mode support:** The LPI2C master also supports the output-only push-pull function required for I2C ultra-fast mode using the SDA and SCL pins. Support for ultra-fast mode also requires the MCFGR1[IGNACK] bit to be set.
- **Push-pull 2-wire support:** A push-pull 2-wire configuration is also available to the LPI2C master that may support a partial high speed mode, if the LPI2C is the only master and all I2C pins on the bus are at the same voltage. This configures the SCL pin as push-pull for every clock except the 9th clock pulse, to allow high speed mode compatible slaves to perform clock stretching. In this mode, the SDA pin is tristated for master-receive data bits and master-transmit ACK/NACK bits, and is configured as push-pull at other times. To avoid the risk of contention when SDA is push-pull, the pin can be configured for open-drain operation, as part of the device-specific configuration.
- **Push-pull 4-wire support:** The push-pull 4-wire configuration separates the SCL input data and output data into separate pins, and separates the SDA input data and output data into separate pins. The SCL/SDA pins are used for input data; the SCLS/SDAS pins are used for output data, with configurable polarity. This simplifies external connections when connecting the LPI2C to the I2C bus through external level shifters or discrete components. When using this 4-wire configuration, the LPI2C master logic and LPI2C slave logic are not able to connect to separate I2C buses.

68.3.2 Slave Mode

To perform all slave mode transfers on the I2C bus, the LPI2C slave logic operates independently from the LPI2C master logic.

Address Matching

The LPI2C slave can be configured:

- To match one of two addresses, using either 7-bit or 10-bit addressing modes for each address
- To match a range of addresses in either 7-bit or 10-bit addressing modes
- To match the General Call Address, and generate appropriate flags
- To match the SMBus Alert Address, and generate appropriate flags
- To detect the high speed mode master code, and to disable the digital filters and output valid delay time until the next STOP condition is detected

After a valid address is matched, the LPI2C slave automatically performs slave-transmit or slave-receive transfers until:

- A NACK is detected (unless SCFGR1[IGNACK] is set)
- A bit error is detected (the LPI2C slave is driving SDA, but a different value is sampled)
- A (repeated) START or STOP condition is detected

Transmit and Receive Data

- The Transmit and Receive Data registers are double-buffered and only update during a slave-transmit and slave-receive transfer, respectively.
- The slave address *that was received* can be configured to be read from either the Receive Data register (for example, when using DMA to transfer data), or from the Address Status register.
- The Transmit Data register can be configured to only request data after a slave-transmit transfer is detected, or to request new data whenever the Transmit Data register is empty.

- The Transmit Data register should only be written when the Transmit Data flag is set.
- The Receive Data register should only be read when the Received Data flag is set (or the Address Valid flag is set and SCFGR1[RXCFG] = 1).
- The Address Status register should only be read when the Address Valid flag is set.

Clock Stretching

The LPI2C slave supports many configurable options for when clock stretching is performed. The following conditions can be configured to perform clock stretching:

- During the 9th clock pulse of the address byte and the Address Valid flag is set.
- During the 9th clock pulse of a slave-transmit transfer and the Transmit Data flag is set.
- During the 9th clock pulse of a slave-receive transfer and the Receive Data flag is set.
- During the 8th clock pulse of an address byte or a slave-receive transfer and the Transmit ACK flag is set. In high speed mode, this is disabled.
- Clock stretching can also be extended for CLKHOLD cycles, to allow additional setup time to sample the SDA pin externally. In high speed mode, this is disabled.

Unless extended by the CLKHOLD configuration, clock stretching extends for one peripheral bus clock cycle after SDA updates when clock stretching is enabled.

Timing Parameters

The LPI2C slave can configure the following timing parameters. These parameters are disabled when SCR[FILTEN] is clear, when SCR[FILTDZ] is set in Doze mode, and when LPI2C slave detects high speed mode. When disabled, the LPI2C slave is clocked directly from the I2C bus, and may not satisfy all timing requirements of the I2C specification (such as SDA minimum hold time in Standard/Fast mode).

- SDA data valid time from SCL negation to SDA update
- SCL hold time when clock stretching is enabled to increase setup time when sampling SDA externally
- SCL glitch filter time
- SDA glitch filter time

The LPI2C slave imposes the following restrictions on the timing parameters.

- FILTSDA must be configured to greater than or equal to FILTSCL (unless compensating for board level skew between SDA and SCL).
- DATAVD must be configured less than the minimum SCL low period.

Error Conditions

The LPI2C slave can detect the following error conditions:

- Bit error flag sets when the LPI2C slave is driving SDA, but samples a different value than what is expected.
- FIFO error flag sets due to a transmit data underrun or a receive data overrun. To eliminate the possibility of underrun and overrun occurring, enable clock stretching.
- FIFO error flag also sets due to an address overrun when SCFGR1[RXCFG] is set, otherwise an address overrun is not flagged. To eliminate the possibility of overrun occurring, enable clock stretching.

The LPI2C slave does not implement a timeout due to SCL and/or SDA being stuck low. If this detection is required, then the LPI2C master logic should be used and so software can reset the LPI2C slave when this condition is detected.

68.3.3 Low power modes

Table 381. LPI2C low power modes

Mode	LPI2C Operation
Run	Normal operations
Stop	Can continue operating in Stop mode if the Doze Enable bit is clear (MCR[DOZEN] = 0) and LPI2C uses an external or internal clock source that remains operating during Stop mode.

68.3.4 Debug mode

Table 382. LPI2C Debug mode

Mode	LPI2C Operation
Debug	Can continue operating in Debug mode if the Debug Enable bit is set (MCR[DBGEN] = 1).

68.3.5 Interrupts and DMA requests

Depending on the specific device, interrupts and DMA requests can be combined in some ways:

- The LPI2C master and slave interrupts may be combined
- The LPI2C master and slave transmit DMA requests may be combined
- The LPI2C master and slave receive DMA requests may be combined

Master mode

The next table lists the master mode sources that can generate LPI2C master interrupts and LPI2C master transmit/receive DMA requests.

Table 383. Master Interrupts and DMA Requests

Master Status (MSR)		Description	Can generate		
Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
TDF	Transmit Data Flag	Data can be written to Transmit FIFO, as configured by the Transmit FIFO Watermark MFCR[TXWATER]	Y	TX	Y
RDF	Receive Data Flag	Data can be read from the Receive FIFO, as configured by the Receive FIFO Watermark MFCR[RXWATER]	Y	RX	Y
EPF	End Packet Flag	Master has transmitted a Repeated START or STOP condition	Y	N	Y
SDF	STOP Detect Flag	Master has transmitted a STOP condition	Y	N	Y

Table continues on the next page...

Table 383. Master Interrupts and DMA Requests (continued)

Master Status (MSR)		Description	Can generate		
Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
NDF	NACK Detect Flag	• During an address byte, the master expected an ACK but detected a NACK • During an address byte, the master expected a NACK but detected an ACK • During a master-transmitter data byte, the master detected a NACK	Y	N	Y
ALF	Arbitration Lost Flag	• The master lost arbitration due to a START/STOP condition detected at the wrong time • Or the master was transmitting data but received different data than the data that was transmitted	Y	N	Y
FEF	FIFO Error Flag	The master is expecting a START condition in the Command FIFO, but the next entry in the Command FIFO is not a START condition	Y	N	Y
PLTF	Pin Low Timeout Flag	Pin low timeout is enabled and SCL (or SDA if configured) is low for longer than the configured timeout	Y	N	Y
DMF	Data Match Flag	The received data matches the configured data match, but the received data is not discarded due to a command FIFO entry	Y	N	Y
MBF	Master Busy Flag	LPI2C master is busy transmitting/receiving data	N	N	N
BBF	Bus Busy Flag	LPI2C master is enabled and activity is detected on the I2C bus, but a STOP condition has not been detected and a bus idle timeout (if enabled) has not occurred.	N	N	N

Slave mode

The next table lists the slave mode sources that can generate LPI2C slave interrupts and the LPI2C slave transmit/receive DMA requests.

Table 384. Slave Interrupts and DMA Requests

Slave Status (SSR)		Description	Can generate		
Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
TDF	Transmit Data Flag	Data can be written to the Slave Transmit Data (STDR)	Y	TX	Y

Table continues on the next page...

Table 384. Slave Interrupts and DMA Requests (continued)

Slave Status (SSR)		Description	Can generate		
Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
RDF	Receive Data Flag	Data can be read from the Slave Receive Data (SRDR)	Y	RX	Y
AVF	Address Valid Flag	Address can be read from the Slave Address Status (SASR)	Y	RX	Y
TAF	Transmit ACK Flag	ACK/NACK can be written to the Slave Transmit ACK (STAR)	Y	N	Y
RSF	Repeated Start Flag	Slave has detected an address match followed by a Repeated START condition	Y	N	Y
SDF	STOP Detect Flag	Slave has detected an address match followed by a STOP condition	Y	N	Y
BEF	Bit Error Flag	Slave was transmitting data, but received different data than what was transmitted	Y	N	Y
FEF	FIFO Error Flag	• Transmit data underrun • Receive data overrun • Address status overrun (when Receive Data Configuration SCFGR1[RXCFG] = 1). FEF flag can only set when clock stretching is disabled.	Y	N	Y
AM0F	Address Match 0 Flag	Slave detected an address match with SAMR[ADDR0] field	Y	N	N
AM1F	Address Match 1 Flag	Slave detected an address match with SAMR[ADDR1] field or using an address range	Y	N	N
GCF	General Call Flag	Slave detected an address match with the General Call address	Y	N	N
SARF	SMBus Alert Response Flag	Slave detected an address match with the SMBus Alert address	Y	N	N
SBF	Slave Busy Flag	LPI2C slave is busy receiving an address byte or is transmitting/receiving data	N	N	N
BBF	Bus Busy Flag	LPI2C slave is enabled and a START condition is detected on I2C bus, but a STOP condition has not been detected	N	N	N

68.3.6 Clocks

Table 385. LPI2C clocks

LPI2C Functional clock	The LPI2C functional clock is asynchronous to the bus clock and can remain enabled in low power modes to support I2C bus transfers by the LPI2C master. The functional clock is also used by the LPI2C slave to support digital filter and data hold time configurations. The LPI2C master divides the functional clock by a prescaler and the resulting frequency must be at least 8 times faster than the I2C bus bandwidth.
External clock	The LPI2C slave logic is clocked directly from the external pins SCL and SDA (or SCLS and SDAS if master and slave are implemented on separate pins). This allows the LPI2C slave to remain operational, even when the LPI2C functional clock is disabled. **NOTE** The LPI2C slave digital filter must be disabled if the LPI2C functional clock is disabled, and this can affect compliance with some of the timing parameters of the I2C specification, such as the data hold time.
Bus clock	The bus clock is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPI2C master and slave registers.

For device-specific clocking information, refer to the Clocking chapter.

68.3.7 Resets

Table 386. LPI2C Resets

Chip reset	The logic and registers for the LPI2C master and slave are reset to their default state on a chip reset.
Software reset	• The LPI2C master implements a software reset bit in its Control Register. The MCR[RST] bit resets all master logic and registers to their default state, except for the MCR register itself. • The LPI2C slave implements a software reset bit in its Control Register. The SCR[RST] bit resets all slave logic and registers to their default state, except for the SCR register itself.
FIFO reset	• The LPI2C master implements write-only control bits that reset the transmit FIFO (MCR[RTF]) and receive FIFO (MCR[RRF]). After a FIFO is reset, that FIFO is empty. • The LPI2C slave implements write-only control bits that reset the transmit data register (SCR[RTF] and receive data register (SCR[RRF]). After a data register is reset, that data register is empty.

68.3.8 Peripheral Triggers

The connection of the LPI2C peripheral triggers to other peripherals depend upon the specific device being used.

Table 387. LPI2C Triggers

Trigger	Description
Master Output Trigger	The LPI2C master generates an output trigger that can be connected to other peripherals on the device. The master output trigger asserts on both a Repeated START or STOP condition, and the master output trigger remains asserted for one cycle of the LPI2C functional clock divided by the prescaler.

Table continues on the next page...

Table 387. LPI2C Triggers (continued)

Trigger	Description
Slave Output Trigger	The LPI2C slave generates an output trigger that can be connected to other peripherals on the device. The slave output trigger asserts on both a Repeated START or STOP condition that occurs after a slave address match, and the slave output trigger remains asserted until the next slave SCL pin negation.
Input Trigger	To control the start of a LPI2C bus transfer, the LPI2C input trigger can be selected instead of the HREQ input. The input trigger is synchronized and to be detected, the input trigger must assert for at least 2 cycles of the LPI2C functional clock divided by the PRESCALE configuration. When the LPI2C is busy, the HREQ input (and therefore the input trigger) is ignored.

68.4 Signal descriptions

Table 388. Signals

Signal	Name	2-Wire Scheme	4-Wire Scheme	I/O
SCL	LPI2C clock line	SCL	In 4-wire mode, this is the SCL input pin.	I/O
SDA	LPI2C data line	SDA	In 4-wire mode, this is the SDA input pin.	I/O
SCLS	Secondary I2C clock line	Not used	In 4-wire mode, this is the SCLS output pin. If LPI2C master/slave are configured to use separate pins, then this is the LPI2C slave SCL pin.	I/O
SDAS	Secondary I2C data line	Not used	In 4-wire mode, this is the SDAS output pin. If LPI2C master/slave are configured to use separate pins, then this is the LPI2C slave SDA pin.	I/O
HREQ	Host request	If host request is asserted and the I2C bus is idle, then it initiates an LPI2C master transfer. HREQ is an additional pin separate from the 2-wire or 4-wire scheme.		I

The I^2C 2-wire serial bus diagram shows the 2 signal connection.

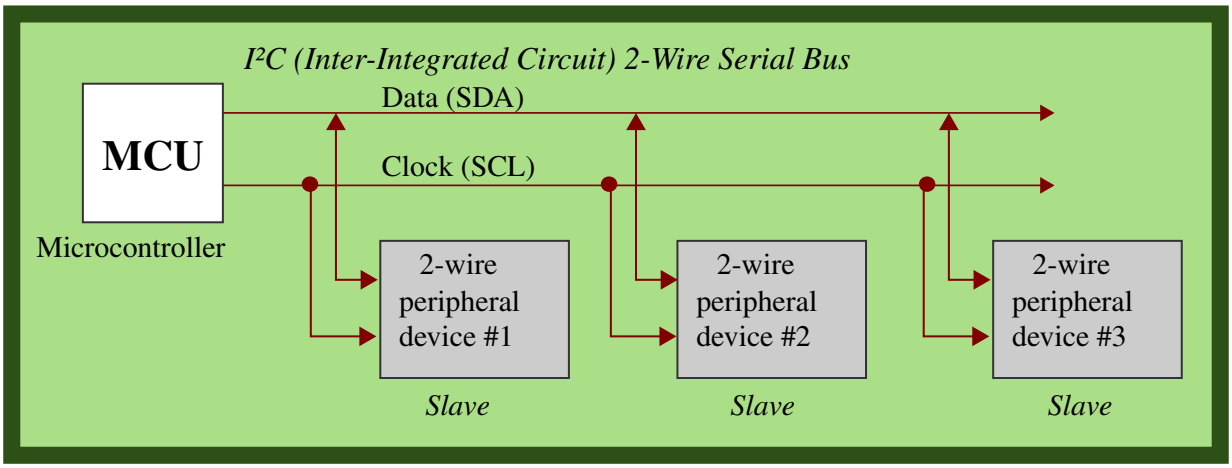


Figure 365. I²C 2-wire serial bus

The I²C 4-wire serial bus diagram shows a possible 4 signal connection.

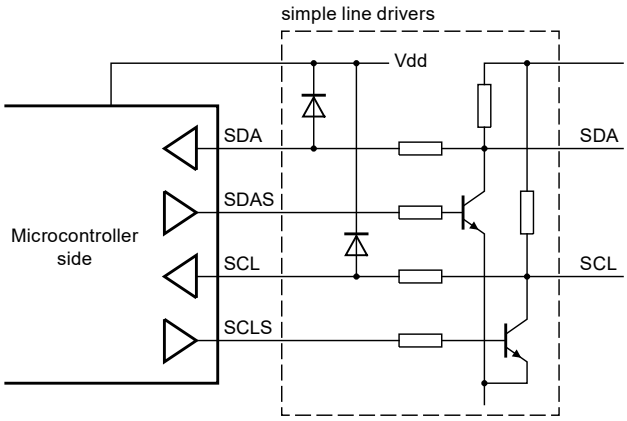


Figure 366. I²C 4-wire serial bus

68.5 Memory Map and Registers

NOTE

Writing a Read-Only (RO) register or reading a Write-Only (WO) register can cause bus errors. This module will not check if programmed values in the registers are correct; the application software must ensure that valid programmed values are being written.

68.5.1 LPI2C register descriptions

68.5.1.1 LPI2C memory map

LPI2C_0 base address: 4035_0000h

LPI2C_1 base address: 4035_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0102_0003h
4h	Parameter (PARAM)	32	R	0000_0202h
10h	Master Control (MCR)	32	RW	0000_0000h
14h	Master Status (MSR)	32	RW	0000_0001h
18h	Master Interrupt Enable (MIER)	32	RW	0000_0000h
1Ch	Master DMA Enable (MDER)	32	RW	0000_0000h
20h	Master Configuration 0 (MCFGR0)	32	RW	0000_0000h
24h	Master Configuration 1 (MCFGR1)	32	RW	0000_0000h
28h	Master Configuration 2 (MCFGR2)	32	RW	0000_0000h
2Ch	Master Configuration 3 (MCFGR3)	32	RW	0000_0000h
40h	Master Data Match (MDMR)	32	RW	0000_0000h
48h	Master Clock Configuration 0 (MCCR0)	32	RW	0000_0000h
50h	Master Clock Configuration 1 (MCCR1)	32	RW	0000_0000h
58h	Master FIFO Control (MFCR)	32	RW	0000_0000h
5Ch	Master FIFO Status (MFSR)	32	R	0000_0000h
60h	Master Transmit Data (MTDR)	32	W	0000_0000h
70h	Master Receive Data (MRDR)	32	R	0000_4000h
110h	Slave Control (SCR)	32	RW	0000_0000h
114h	Slave Status (SSR)	32	RW	0000_0000h
118h	Slave Interrupt Enable (SIER)	32	RW	0000_0000h
11Ch	Slave DMA Enable (SDER)	32	RW	0000_0000h
124h	Slave Configuration 1 (SCFGR1)	32	RW	0000_0000h
128h	Slave Configuration 2 (SCFGR2)	32	RW	0000_0000h
140h	Slave Address Match (SAMR)	32	RW	0000_0000h
150h	Slave Address Status (SASR)	32	R	0000_4000h
154h	Slave Transmit ACK (STAR)	32	RW	0000_0000h
160h	Slave Transmit Data (STDR)	32	W	0000_0000h
170h	Slave Receive Data (SRDR)	32	R	0000_4000h

68.5.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram

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

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

Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module design specification. Read-only field.
23-16 MINOR	Minor Version Number Returns the minor version number for the module design specification. Read-only field.
15-0 FEATURE	Feature Specification Number Returns the feature set number. Read-only field. 0000_0000_0000_0010b - Master only, with standard feature set 0000_0000_0000_0011b - Master and slave, with standard feature set

68.5.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Contains parameter values that were implemented in the module.

Diagram

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

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

Fields

Field	Function
31-16 —	Reserved
15-12 —	Reserved
11-8 MRXFIFO	Master Receive FIFO Size Configures the number of words in the master receive FIFO to $2^{MRXFIFO}$
7-4 —	Reserved
3-0 MTXFIFO	Master Transmit FIFO Size Configures the number of words in the master transmit FIFO to $2^{MTXFIFO}$

68.5.1.4 Master Control (MCR)

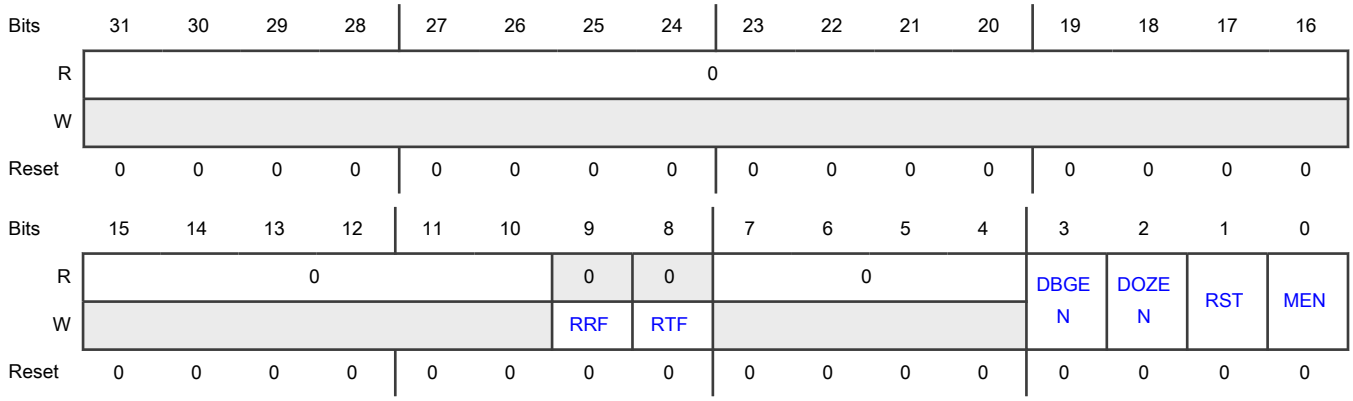
Offset

Register	Offset
MCR	10h

Function

The MCR register contains resets, debug enable, and other master control settings.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Resets the Receive FIFO. 0b - No effect 1b - Receive FIFO is reset
8 RTF	Reset Transmit FIFO Resets the Transmit FIFO. 0b - No effect 1b - Transmit FIFO is reset
7-4 —	Reserved
3 DBGEN	Debug Enable Enables the Master in Debug mode. 0b - Master is disabled in debug mode 1b - Master is enabled in debug mode
2 DOZEN	Doze mode enable Enables or disables the master in Doze mode. 0b - Master is enabled in Doze mode 1b - Master is disabled in Doze mode
1	Software Reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
RST	Resets all internal master logic and registers, except the Master Control (MCR) register. RST remains set until cleared by software. The reset takes effect immediately and remains asserted until negated by software. There is no minimum delay required before clearing the software reset. 0b - Master logic is not reset 1b - Master logic is reset
0 MEN	Master Enable Enables the Master logic. 0b - Master logic is disabled 1b - Master logic is enabled

68.5.1.5 Master Status (MSR)

Offset

Register	Offset
MSR	14h

Function

MSR contains status flags for transmit and receive data; Start and Stop conditions; and bus and master busy or idle status.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	DMF	PLTF	FEF	ALF	NDF	SDF	EPF	0						RDF	TDF
W		W1C	W1C	W1C	W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-26 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
25 BBF	Bus Busy Flag Indicates that the I2C bus is busy. 0b - I2C Bus is idle 1b - I2C Bus is busy
24 MBF	Master Busy Flag Indicates that the I2C Master is busy. 0b - I2C Master is idle 1b - I2C Master is busy
23-16 —	Reserved
15 —	Reserved
14 DMF	Data Match Flag Indicates that the received data has matched the MDMR[MATCH0] and/or MDMR[MATCH1] fields (as configured by MCFGR1[MATCFG]. Received data *that is discarded due to CMD field* does not cause DMF to set. 0b - Have not received matching data 1b - Have received matching data
13 PLTF	Pin Low Timeout Flag Sets when the SCL and/or SDA input is low for more than PINLOW cycles (Pin Low Timeout, MCFGR3[PINLOW]), even when the LPI2C master is idle. • Software is responsible for resolving the pin low condition. • PLTF cannot be cleared as long as the pin low timeout continues. • Before the LPI2C can initiate a START condition, the PLTF must be cleared. 0b - Pin low timeout has not occurred or is disabled 1b - Pin low timeout has occurred
12 FEF	FIFO Error Flag Detects an attempt to send or receive data without first generating a (repeated) START condition. This can occur if the transmit FIFO underflows when MCFGR1[AUTOSTOP] = 1. When FEF is set, the LPI2C master sends a STOP condition (if busy), and does not initiate a new START condition until FEF has been cleared. 0b - No error 1b - Master sending or receiving data without a START condition

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 ALF	Arbitration Lost Flag Set if either of these conditions exist: • The LPI2C master transmits a logic one and detects a logic zero on the I2C bus • The LPI2C master detects a START or STOP condition while the LPI2C master is transmitting data When ALF sets, the LPI2C master releases the I2C bus (goes idle), and the LPI2C master does not initiate a new START condition until the ALF has been cleared. 0b - Master has not lost arbitration 1b - Master has lost arbitration
10 NDF	NACK Detect Flag Set if the LPI2C master detects a NACK it was not expecting when transmitting an address or data. When set, the master does not initiate a new START condition until NDF has been cleared. If a NACK is expected for a given address (as configured by the command word), then the NDF sets if a NACK is not generated. When NDF is set, the LPI2C master automatically transmits a STOP condition if MCFGR1[AUTOSTOP] = 1, or the transmit FIFO is not empty. 0b - Unexpected NACK was not detected 1b - Unexpected NACK was detected
9 SDF	STOP Detect Flag Set when the LPI2C master generates a STOP condition. 0b - Master has not generated a STOP condition 1b - Master has generated a STOP condition
8 EPF	End Packet Flag Set when the LPI2C master generates either a repeated START condition or a STOP condition. Does not set when the master first generates a START condition. 0b - Master has not generated a STOP or Repeated START condition 1b - Master has generated a STOP or Repeated START condition
7-2 —	Reserved
1 RDF	Receive Data Flag RDF sets whenever the number of words in the receive FIFO is greater than MFCR[RXWATER]. 0b - Receive Data is not ready 1b - Receive data is ready
0 TDF	Transmit Data Flag TDF sets whenever the number of words in the transmit FIFO is equal or less than MFCR[TXWATER].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Transmit data is not requested 1b - Transmit data is requested

68.5.1.6 Master Interrupt Enable (MIER)

Offset

Register	Offset
MIER	18h

Function

MIER contains transmit and receive data interrupt enables; Start, Stop, and Nack detect interrupt enables; and DMA interrupt enables.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	DMIE	PLTIE	FEIE	ALIE	NDIE	SDIE	EPIE	0						RDIE	TDIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 —	Reserved
14 DMIE	Data Match Interrupt Enable Enables interrupt for data match. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
13 PLTIE	Pin Low Timeout Interrupt Enable Enables interrupt for pin low timeout. 0b - Disabled 1b - Enabled
12 FEIE	FIFO Error Interrupt Enable Enables interrupt for FIFO error. 0b - Disabled 1b - Enabled
11 ALIE	Arbitration Lost Interrupt Enable Enables interrupt for arbitration lost. 0b - Disabled 1b - Enabled
10 NDIE	NACK Detect Interrupt Enable Enables interrupt for NACK detect. 0b - Disabled 1b - Enabled
9 SDIE	STOP Detect Interrupt Enable Enables interrupt for STOP detect. 0b - Disabled 1b - Enabled
8 EPIE	End Packet Interrupt Enable Enables interrupt for end packet. 0b - Disabled 1b - Enabled
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable Enables interrupt for receive data. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 TDIE	Transmit Data Interrupt Enable Enables interrupt for transmit data. 0b - Disabled 1b - Enabled

68.5.1.7 Master DMA Enable (MDER)

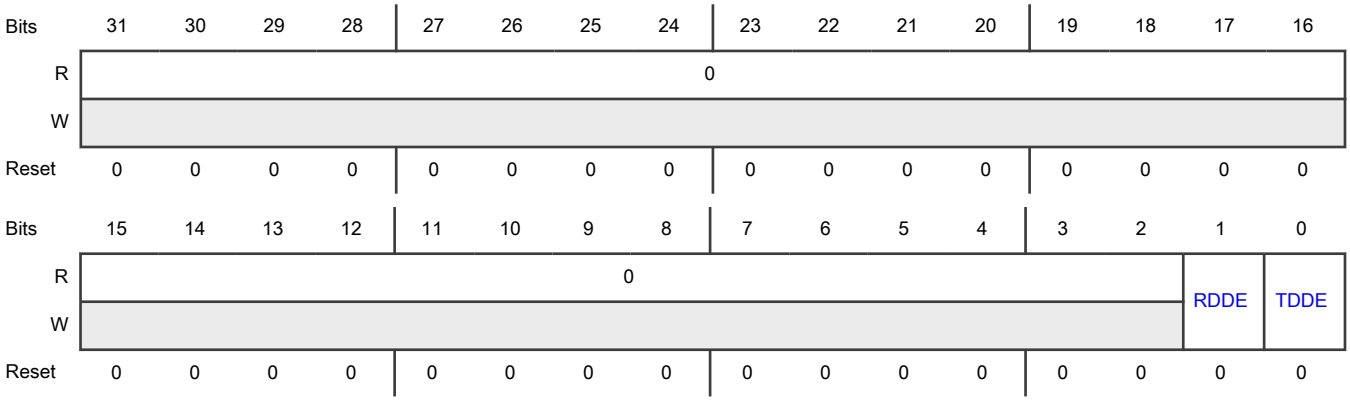
Offset

Register	Offset
MDER	1Ch

Function

MDER contains DMA transmit, request, and receive enables.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RDDE	Receive Data DMA Enable Enables DMA receive data. 0b - DMA request is disabled 1b - DMA request is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 TDDE	Transmit Data DMA Enable Enables DMA transmit data. 0b - DMA request is disabled 1b - DMA request is enabled

68.5.1.8 Master Configuration 0 (MCFGR0)

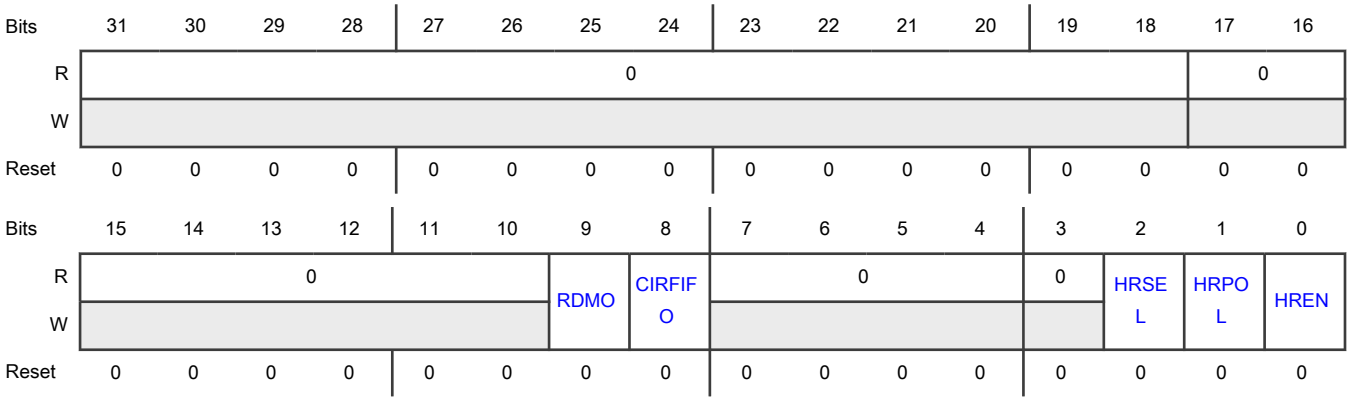
Offset

Register	Offset
MCFGR0	20h

Function

MCFGR0 contains host settings and other receive and transfer settings.

Diagram



Fields

Field	Function
31-18 —	Reserved
17-16 —	Reserved
15-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 RDMO	Receive Data Match Only When enabled, all received data that does not cause the Data Match Flag (MSR[DMF]) to set is discarded. After MSR[DMF] is set, the RDMO configuration is ignored. When disabling RDMO, clear RDMO before clearing MSR[DMF], to ensure that no receive data is lost. 0b - Received data is stored in the receive FIFO 1b - Received data is discarded unless the the Data Match Flag (MSR[DMF]) is set
8 CIRFIFO	Circular FIFO Enable When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO empties as normal, but after the LPI2C master is idle and the transmit FIFO is empty, then the read pointer value will be restored from the temporary register. This causes the contents of the transmit FIFO to be cycled through repeatedly. If MCFGR1[AUTOSTOP] is set, then a STOP condition is sent whenever the transmit FIFO is empty and the read pointer is restored. 0b - Circular FIFO is disabled 1b - Circular FIFO is enabled
7-4 —	Reserved
3 —	Reserved
2 HRSEL	Host Request Select Selects the source of the host request input. When host request is enabled, HRSEL should remain static (the Host Request Select should not change). 0b - Host request input is pin HREQ 1b - Host request input is input trigger
1 HRPOL	Host Request Polarity Configures the polarity of the host request pin. When host request is enabled, HRPOL should remain static (Host Request Polarity should not change). 0b - Active low 1b - Active high
0 HREN	Host Request Enable When enabled, the LPI2C master only initiates a START condition if the host request input is asserted and the bus is idle. A repeated START is not affected by the host request. 0b - Host request input is disabled 1b - Host request input is enabled

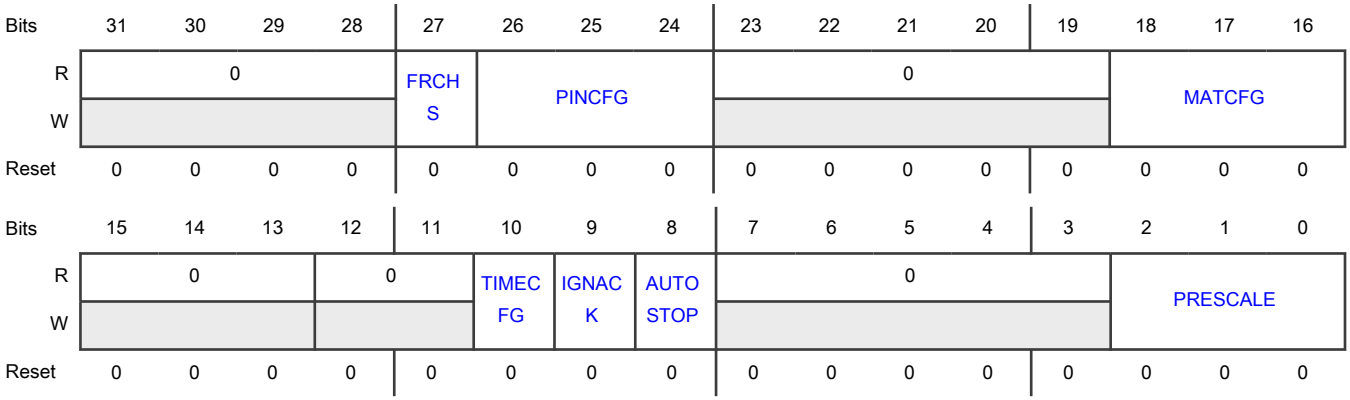
68.5.1.9 Master Configuration 1 (MCFGR1)

Offset

Register	Offset
MCFGR1	24h

Function
The MCFGR1 should only be written when the I2C Master is disabled.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 FRCHS	Force HS-mode When set, forces the LPI2C pin state into HS-mode. Setting this does not impact the digital filter configuration or digital timing parameters. 0b - No effect 1b - LPI2C pin state forced into HS-mode.
26-24 PINCFG	Pin Configuration Configures the pin mode for LPI2C.

Table 389. 2-pin / 4-pin pin configurations for masters and slaves

PINCFG	SCL / SDA pins	SCLS / SDAS pins
000	Bi-directional open drain for master and slave	Not used
001	Output-only (ultra-fast mode) open drain for master and slave	Not used
010	Bi-directional push-pull for master and slave	Not used

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>Table 389. 2-pin / 4-pin pin configurations for masters and slaves (continued)
<table>
<tr><th>PINCFG</th><th>SCL / SDA pins</th><th>SCLS / SDAS pins</th></tr>
<tr><td>011</td><td>Input only for master and slave</td><td>Output-only push-pull for master and slave</td></tr>
<tr><td>100</td><td>Bi-directional open drain for master</td><td>Bi-directional open drain for slave</td></tr>
<tr><td>101</td><td>Output-only (ultra-fast mode) open drain for master</td><td>Output-only open drain for slave</td></tr>
<tr><td>110</td><td>Bi-directional push-pull for master</td><td>Bi-directional push-pull for slave</td></tr>
<tr><td>111</td><td>Input only for master and slave</td><td>Inverted output-only push-pull for master and slave</td></tr>
</table>
000b - 2-pin open drain mode
001b - 2-pin output only mode (ultra-fast mode)
010b - 2-pin push-pull mode
011b - 4-pin push-pull mode
100b - 2-pin open drain mode with separate LPI2C slave
101b - 2-pin output only mode (ultra-fast mode) with separate LPI2C slave
110b - 2-pin push-pull mode with separate LPI2C slave
111b - 4-pin push-pull mode (inverted outputs)</td></tr>
<tr><td>23-19
—</td><td>Reserved</td></tr>
<tr><td>18-16
MATCFG</td><td>Match Configuration
Configures the condition that causes MSR[DMF] to set. See Master Data Match (MDMR).
000b - Match is disabled
001b - Reserved
010b - Match is enabled (1st data word equals MDMR[MATCH0] OR MDMR[MATCH1])
011b - Match is enabled (any data word equals MDMR[MATCH0] OR MDMR[MATCH1])
100b - Match is enabled (1st data word equals MDMR[MATCH0] AND 2nd data word equals MDMR[MATCH1)
101b - Match is enabled (any data word equals MDMR[MATCH0] AND next data word equals MDMR[MATCH1)
110b - Match is enabled (1st data word AND MDMR[MATCH1] equals MDMR[MATCH0] AND MDMR[MATCH1])
111b - Match is enabled (any data word AND MDMR[MATCH1] equals MDMR[MATCH0] AND MDMR[MATCH1])</td></tr>
<tr><td>15-13
—</td><td>Reserved</td></tr>
</table>

Table continues on the next page...

Field	Function
12-11 —	Reserved
10 TIMECFG	Timeout Configuration TIMECFG configures the timeout settings for the Pin Low Timeout Flag field (MSR[PLTF]). 0b - MSR[PLTF] sets if SCL is low for longer than the configured timeout 1b - MSR[PLTF] sets if either SCL or SDA is low for longer than the configured timeout
9 IGNACK	IGNACK When set, the received NACK field is ignored and assumed to be ACK. IGNACK bit is required to be set in Ultra-Fast Mode. 0b - LPI2C Master receives ACK and NACK normally 1b - LPI2C Master treats a received NACK as if it (NACK) was an ACK
8 AUTOSTOP	Automatic STOP Generation When enabled, a STOP condition is generated whenever the LPI2C master is busy and the transmit FIFO is empty. The STOP condition can also be generated using a transmit FIFO command. 0b - No effect 1b - STOP condition is automatically generated whenever the transmit FIFO is empty and the LPI2C master is busy
7-3 —	Reserved
2-0 PRESCALE	Prescaler Configures the clock prescaler used for all LPI2C master logic, except for the digital glitch filters. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

68.5.1.10 Master Configuration 2 (MCFGR2)

Offset

Register	Offset
MCFGR2	28h

Function

The Master Configuration Register 2 should only be written when the I2C Master is disabled.

Diagram

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

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

Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch Filter SDA Configures the I2C master digital glitch filters for SDA input. • A configuration of 0 disables the glitch filter • Glitches equal to or less than FILTSDA cycles long are filtered out and ignored • The latency through the glitch filter is equal to FILTSDA cycles, and must be configured to be less than the minimum SCL low or high period • The glitch filter cycle count is not affected by the PRESCALE configuration, and the glitch filter cycle count is automatically bypassed in High Speed mode
23-20 —	Reserved
19-16 FILTSCL	Glitch Filter SCL Configures the I2C master digital glitch filters for SCL input. • A configuration of 0 disables the glitch filter • Glitches equal to or less than FILTSCL cycles long are filtered out and ignored.The FILTSCL cycles are based on the functional clock. • The latency through the glitch filter is equal to FILTSCL cycles, and must be configured to be less than the minimum SCL low or high period • The glitch filter cycle count is not affected by the PRESCALE configuration, and the glitch filter cycle count is automatically bypassed in High Speed mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11-0 BUSIDLE	Bus Idle Timeout Configures the bus idle timeout period in clock cycles. • If both SCL and SDA are high for longer than BUSIDLE cycles, then the I2C bus is assumed to be idle and the master can generate a START condition • When Bus Idle Timeout is set to zero, the Bus Idle Timeout is disabled

68.5.1.11 Master Configuration 3 (MCFGR3)

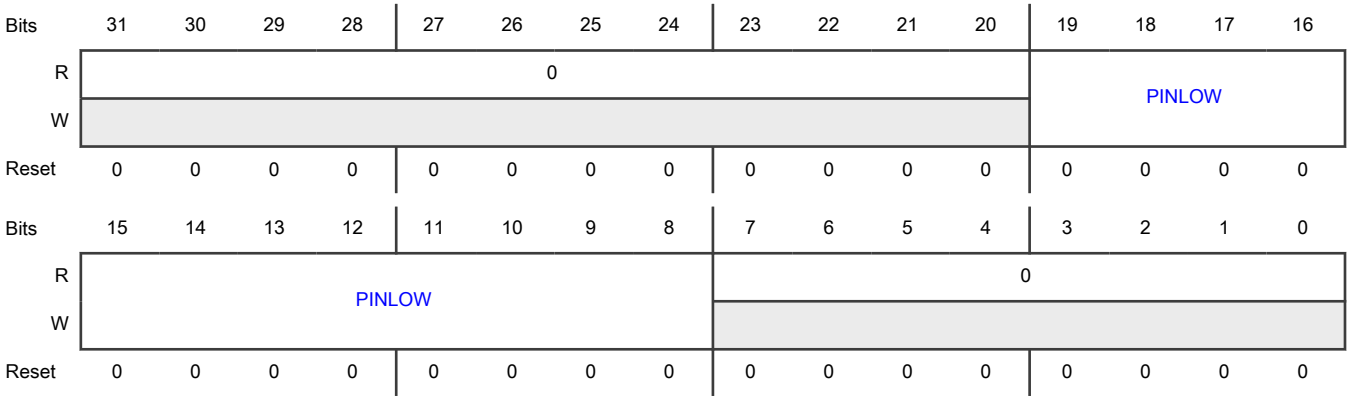
Offset

Register	Offset
MCFGR3	2Ch

Function

The MCFGR3 should only be written when the I2C Master is disabled.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-8	Pin Low Timeout

Table continues on the next page...

Table continued from the previous page...

Field	Function
PINLOW	Configures the pin low timeout flag in clock cycles. • If SCL or, either SCL or SDA, is low for longer than (PINLOW * 256) cycles, then MSR[PLTF] is set. • When Pin Low Timeout is set to zero, the Pin Low Timeout feature is disabled
7-0 —	Reserved

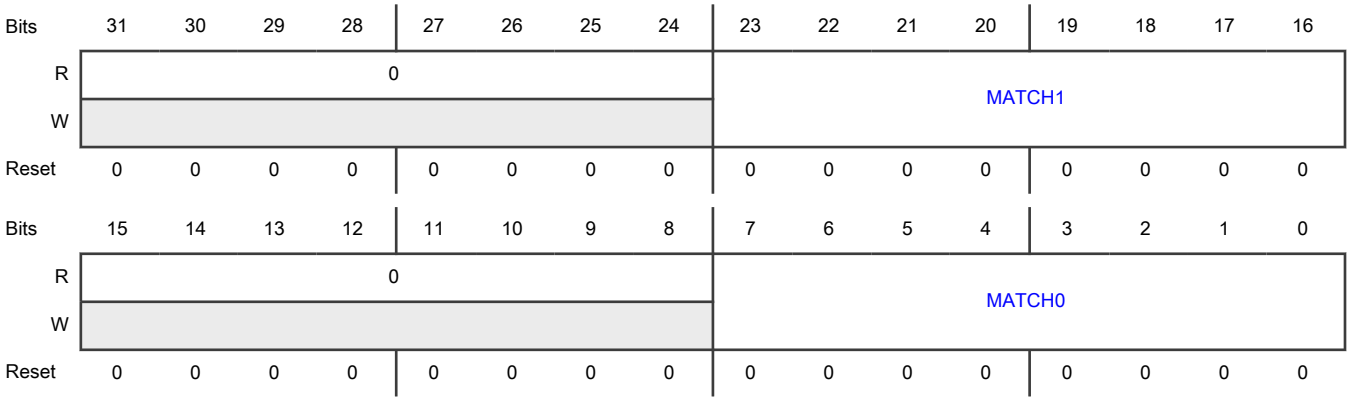
68.5.1.12 Master Data Match (MDMR)

Offset

Register	Offset
MDMR	40h

Function
The MDMR should only be written when the I2C Master is disabled or idle.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 MATCH1	Match 1 Value Compared against the received data when receive data match is enabled.
15-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7-0 MATCH0	Match 0 Value Compared against the received data when receive data match is enabled.

68.5.1.13 Master Clock Configuration 0 (MCCR0)

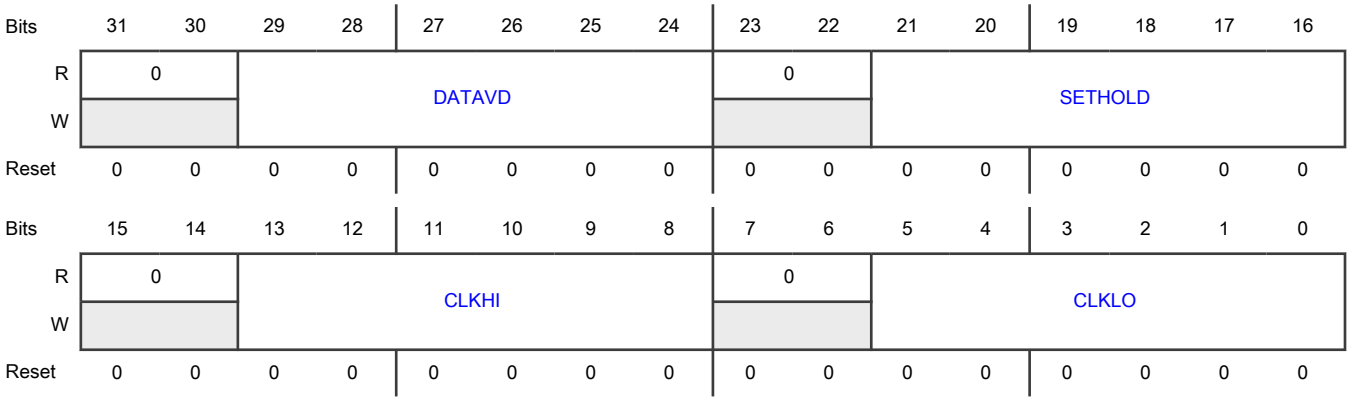
Offset

Register	Offset
MCCR0	48h

Function

The MCCR0 cannot be changed when the I2C master is enabled and is used for standard, fast, fast-mode plus and ultra-fast transfers.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 DATAVD	Data Valid Delay Minimum number of cycles (minus one) that is used as the data hold time for SDA. Must be configured less than the minimum SCL low period.
23-22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
21-16 SETHOLD	Setup Hold Delay Minimum number of cycles (minus one) that is used by the master for these conditions: • Hold time for a START • Setup and hold time for a repeated START • Setup time for a STOP The setup time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
15-14 —	Reserved
13-8 CLKHI	Clock High Period Minimum number of cycles (minus one) that the SCL clock is driven high by the master. The SCL high time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
7-6 —	Reserved
5-0 CLKLO	Clock Low Period Minimum number of cycles (minus one) that the SCL clock is driven low by the master. The Clock Low Period value is also used for the minimum bus free time between a STOP and a START condition; this is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.

68.5.1.14 Master Clock Configuration 1 (MCCR1)

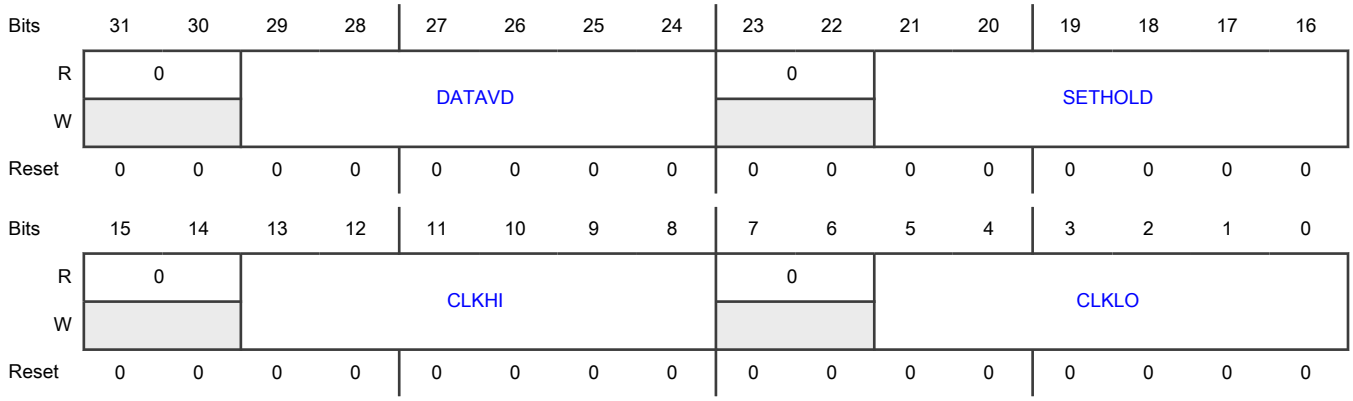
Offset

Register	Offset
MCCR1	50h

Function

The MCCR1 cannot be changed when the I2C master is enabled and is used for high speed mode transfers. The separate clock configuration for high speed mode allows arbitration to take place in Fast mode (with timing configured by Master Clock Configuration 0 (MCCR0)), before switching to high speed mode (with timing configured by MCCR1).

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 DATAVD	Data Valid Delay Minimum number of cycles (minus one) that is used as the data hold time for SDA. The DATAVD must be configured to be less than the minimum SCL low period.
23-22 —	Reserved
21-16 SETHOLD	Setup Hold Delay Minimum number of cycles (minus one) that is used by the master for these conditions: • Hold time for a START condition • Setup and hold time for a repeated START condition • Setup time for a STOP condition The setup time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
15-14 —	Reserved
13-8 CLKHI	Clock High Period Minimum number of cycles (minus one) that the SCL clock is driven high by the master. The SCL high time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
7-6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5-0 CLKLO	Clock Low Period Minimum number of cycles (minus one) that the SCL clock is driven low by the master. The CLKLO value is also used for the minimum bus free time between a STOP and a START condition; this is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.

68.5.1.15 Master FIFO Control (MFCR)

Offset

Register	Offset
MFCR	58h

Function

The Master FIFO control register is only used in Stop mode when the MFCR register is static (i.e., the MFCR register is not changing).

Diagram

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

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

Fields

Field	Function
31-18 —	Reserved
17-16 RXWATER	Receive FIFO Watermark The Receive Data Flag (SSR[RDF]) is set whenever the number of words in the receive FIFO is greater than RXWATER. Writing a value equal to or greater than the FIFO size truncates the value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-2 —	Reserved
1-0 TXWATER	Transmit FIFO Watermark The Transmit Data Flag (SSR[TDF]) is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. Writing a value equal to or greater than the FIFO size truncates the value.

68.5.1.16 Master FIFO Status (MFSR)

Offset

Register	Offset
MFSR	5Ch

Function

MFSR gives the number of words in the transmit and receive FIFO.

Diagram

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

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

Fields

Field	Function
31-19 —	Reserved
18-16 RXCOUNT	Receive FIFO Count Returns the number of words in the receive FIFO.
15-3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2-0 TXCOUNT	Transmit FIFO Count Returns the number of words in the transmit FIFO.

68.5.1.17 Master Transmit Data (MTDR)

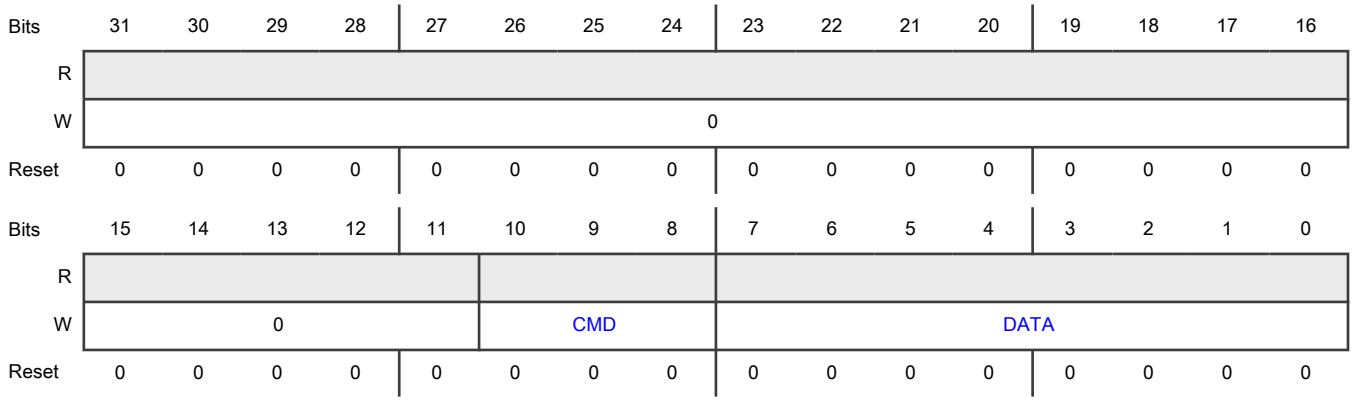
Offset

Register	Offset
MTDR	60h

Function

- An 8-bit write to the CMD field is ignored and does not increment the FIFO write pointer.
- An 8-bit write to the DATA field zero-extends the CMD field and increments the FIFO write pointer.
- A 16-bit or 32-bit writes both the CMD and DATA fields and increments the FIFO write pointer.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 CMD	Command Data 000b - Transmit DATA[7:0]

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - Receive (DATA[7:0] + 1) bytes 010b - Generate STOP condition 011b - Receive and discard (DATA[7:0] + 1) bytes 100b - Generate (repeated) START and transmit address in DATA[7:0] 101b - Generate (repeated) START and transmit address in DATA[7:0]. This transfer expects a NACK to be returned. 110b - Generate (repeated) START and transmit address in DATA[7:0] using high speed mode 111b - Generate (repeated) START and transmit address in DATA[7:0] using high speed mode. This transfer expects a NACK to be returned.
7-0 DATA	Transmit Data Performing an 8-bit write to DATA zero-extends the CMD field.

68.5.1.18 Master Receive Data (MRDR)

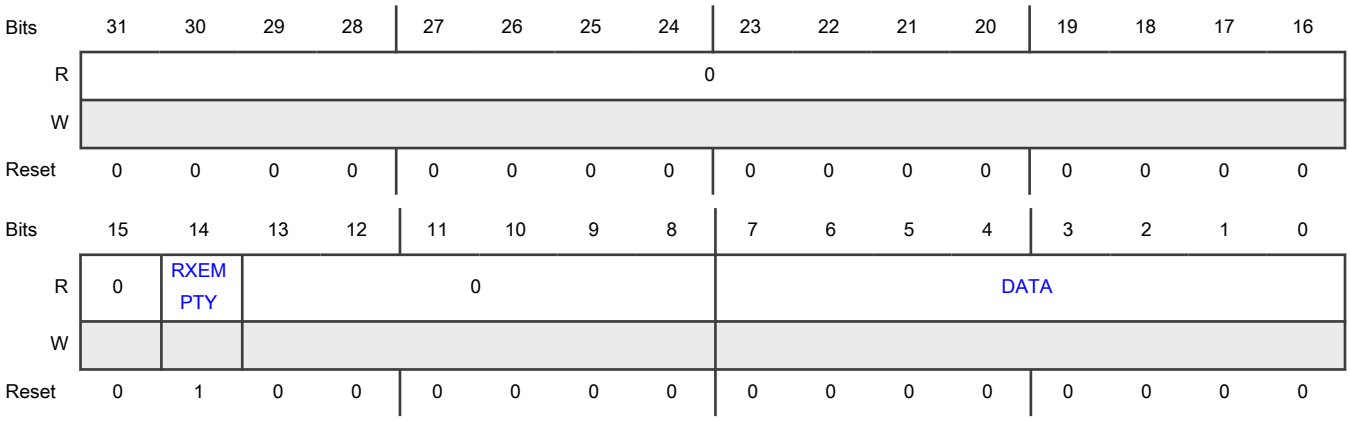
Offset

Register	Offset
MRDR	70h

Function

Reading the Receive Data register returns the data received by the I2C master that has not been discarded.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 RXEMPTY	RX Empty RXEMPTY gives the empty status of the Master receive data FIFO. 0b - Receive FIFO is not empty 1b - Receive FIFO is empty
13-8 —	Reserved
7-0 DATA	Receive Data Receive data can be discarded due to the CMD field, or the master can be configured to discard non-matching data.

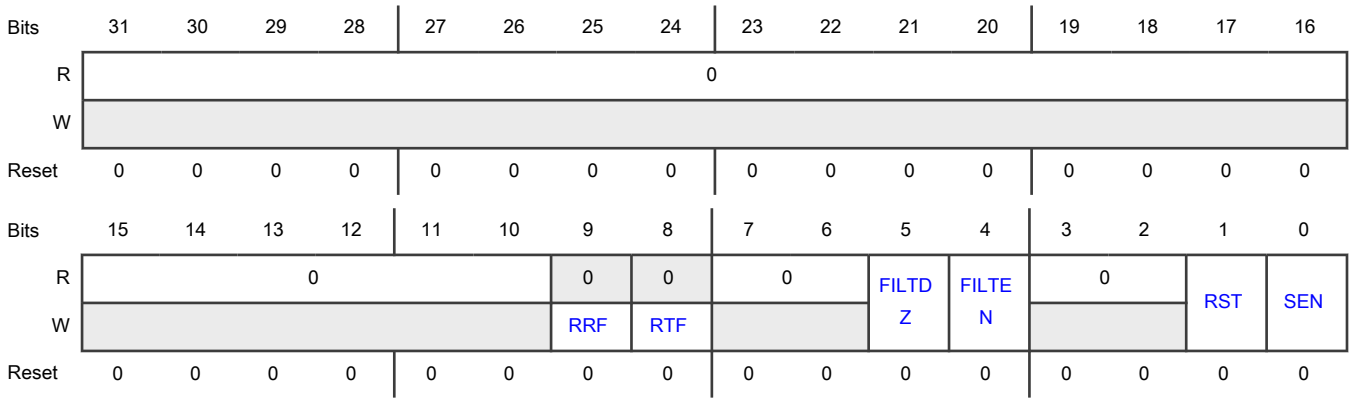
68.5.1.19 Slave Control (SCR)

Offset

Register	Offset
SCR	110h

Function
SCR contains resets and other slave control settings.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Resets the receive FIFO to b be empty. 0b - No effect 1b - Receive Data Register is now empty
8 RTF	Reset Transmit FIFO Resets the transmit FIFO to be empty. 0b - No effect 1b - Transmit Data Register is now empty
7-6 —	Reserved
5 FILTDZ	Filter Doze Enable FILTDZ should only be updated when the I2C Slave is disabled. 0b - Filter remains enabled in Doze mode 1b - Filter is disabled in Doze mode
4 FILTEN	Filter Enable FILTEN should only be updated when the I2C Slave is disabled. 0b - Disable digital filter and output delay counter for slave mode 1b - Enable digital filter and output delay counter for slave mode
3-2 —	Reserved
1 RST	Software Reset The reset takes effect immediately and remains asserted until negated by software. There is no minimum delay required before clearing the software reset. 0b - Slave mode logic is not reset 1b - Slave mode logic is reset
0 SEN	Slave Enable Enables the I2C Slave mode. 0b - I2C Slave mode is disabled 1b - I2C Slave mode is enabled

68.5.1.20 Slave Status (SSR)

Offset

Register	Offset
SSR	114h

Function

SSR contains status flags for transmit and receive data; error conditions; and bus and slave busy or idle status.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SARF	GCF	AM1F	AM0F	FEF	BEF	SDF	RSF	0				TAF	AVF	RDF	TDF
W					W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25 BBF	Bus Busy Flag Indicates if an I2C bus is idle or busy. 0b - I2C Bus is idle 1b - I2C Bus is busy
24 SBF	Slave Busy Flag Indicates if an I2C slave is idle or busy. 0b - I2C Slave is idle 1b - I2C Slave is busy
23-16 —	Reserved
15	SMBus Alert Response Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
SARF	• SARF is cleared by reading the Slave Address Status (SASR) register • SARF cannot generate an asynchronous wakeup 0b - SMBus Alert Response is disabled or not detected 1b - SMBus Alert Response is enabled and detected
14 GCF	General Call Flag Indicates whether a slave has detected the General Call Address. • General Call Flag is cleared by reading the Slave Address Status (SASR) register • General Call Flag cannot generate an asynchronous wakeup 0b - Slave has not detected the General Call Address or the General Call Address is disabled 1b - Slave has detected the General Call Address
13 AM1F	Address Match 1 Flag Indicates that the received address has matched the ADDR1 field or ADDR0 to ADDR1 range as configured by SCFGR1[ADDRCFG]. • Address Match 1 Flag is cleared by reading the Slave Address Status (SASR) register • Address Match 1 Flag cannot generate an asynchronous wakeup 0b - Have not received an ADDR1 or ADDR0/ADDR1 range matching address 1b - Have received an ADDR1 or ADDR0/ADDR1 range matching address
12 AM0F	Address Match 0 Flag Indicates that the received address has matched the ADDR0 field as configured by SCFGR1[ADDRCFG]. • AM0F is cleared by reading the Slave Address Status (SASR) register • AM0F cannot generate an asynchronous wakeup 0b - Have not received an ADDR0 matching address 1b - Have received an ADDR0 matching address
11 FEF	FIFO Error Flag FEF can only be set when clock stretching is disabled. 0b - FIFO underflow or overflow was not detected 1b - FIFO underflow or overflow was detected
10 BEF	Bit Error Flag BEF sets if the LPI2C slave transmits a logic one and detects a logic zero on the I2C bus. The slave ignores the rest of the transfer until the next (repeated) START condition. 0b - Slave has not detected a bit error 1b - Slave has detected a bit error

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 SDF	STOP Detect Flag SDF sets when the LPI2C slave detects a STOP condition and if the LPI2C slave matched the last address byte. 0b - Slave has not detected a STOP condition 1b - Slave has detected a STOP condition
8 RSF	Repeated Start Flag RSF sets when the LPI2C slave detects a repeated START condition and if the LPI2C slave matched the last address byte. The RSF does not set when the slave first detects a START condition. 0b - Slave has not detected a Repeated START condition 1b - Slave has detected a Repeated START condition
7-4 —	Reserved
3 TAF	Transmit ACK Flag TAF is cleared by writing to the Slave Transmit ACK (STAR) register. 0b - Transmit ACK/NACK is not required 1b - Transmit ACK/NACK is required
2 AVF	Address Valid Flag AVF is cleared by reading the Slave Address Status (SASR) register. When SCFGR1[RXCFG] = 1, SSR[AVF] is also cleared by reading the Slave Receive Data (SRDR) register. 0b - Address Status Register is not valid 1b - Address Status Register is valid
1 RDF	Receive Data Flag RDF is cleared by reading the Slave Receive Data (SRDR) register. When SCFGR1[RXCFG] = 1, RDF is not cleared when reading the Slave Receive Data (SRDR) register and if SSR[AVF] is set. 0b - Receive data is not ready 1b - Receive data is ready
0 TDF	Transmit Data Flag TDF is cleared by writing the to the Slave Transmit Data (STDR) register. When SCFGR1[TXCFG] = 0 and if a NACK or a Repeated START or STOP condition is detected, then TDF is also cleared. 0b - Transmit data not requested 1b - Transmit data is requested

68.5.1.21 Slave Interrupt Enable (SIER)

Offset

Register	Offset
SIER	118h

Function

SIER contains transmit and receive data interrupt enables; Start and Stop detect interrupt enables; and other slave interrupt enables.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SARIE	GCIE	AM1IE	AM0IE	FEIE	BEIE	SDIE	RSIE	0				TAIE	AVIE	RDIE	TDIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 SARIE	SMBus Alert Response Interrupt Enable Enables interrupt for SMBus alert response. 0b - Disabled 1b - Enabled
14 GCIE	General Call Interrupt Enable Enables interrupt for general call. 0b - Disabled 1b - Enabled
13 AM1IE	Address Match 1 Interrupt Enable Enables interrupt for address match 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
12 AM0IE	Address Match 0 Interrupt Enable Enables interrupt for address match 0. 0b - Disabled 1b - Enabled
11 FEIE	FIFO Error Interrupt Enable Enables interrupt for FIFO error. 0b - Disabled 1b - Enabled
10 BEIE	Bit Error Interrupt Enable Enables interrupt for bit error. 0b - Disabled 1b - Enabled
9 SDIE	STOP Detect Interrupt Enable Enables interrupt for STOP detect. 0b - Disabled 1b - Enabled
8 RSIE	Repeated Start Interrupt Enable Enables interrupt for repeated start. 0b - Disabled 1b - Enabled
7-4 —	Reserved
3 TAIE	Transmit ACK Interrupt Enable Enables interrupt for transmit ACK. 0b - Disabled 1b - Enabled
2 AVIE	Address Valid Interrupt Enable Enables interrupt for valid address.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
1 RDIE	Receive Data Interrupt Enable Enables interrupt for receive data. 0b - Disabled 1b - Enabled
0 TDIE	Transmit Data Interrupt Enable Enables interrupt for transmit data. 0b - Disabled 1b - Enabled

68.5.1.22 Slave DMA Enable (SDER)

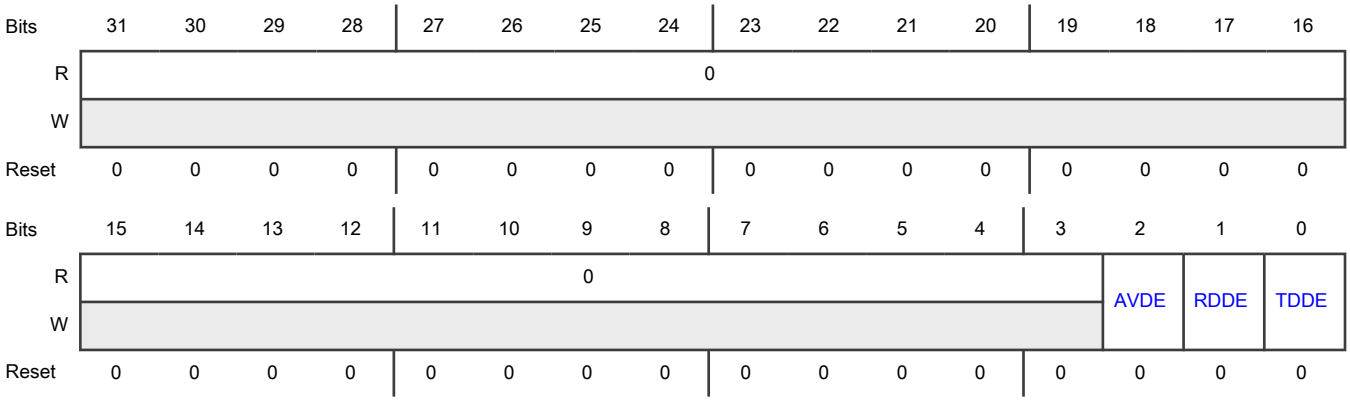
Offset

Register	Offset
SDER	11Ch

Function

SDER contains the transmit, request, and receive enables for DMA.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 AVDE	Address Valid DMA Enable The Address Valid DMA request is shared with the Receive Data DMA request. If both Address Valid DMA request and Receive Data DMA request are enabled, then set SCFGR1[RXCFG] = 1, to allow the DMA to read the address from the Slave Receive Data (SRDR) register. 0b - DMA request is disabled 1b - DMA request is enabled
1 RDDE	Receive Data DMA Enable Enables receive data for DMA. 0b - DMA request is disabled 1b - DMA request is enabled
0 TDDE	Transmit Data DMA Enable Enables transmit data for DMA. 0b - DMA request is disabled 1b - DMA request is enabled

68.5.1.23 Slave Configuration 1 (SCFGR1)

Offset

Register	Offset
SCFGR1	124h

Function
The Slave Configuration Register 1 should only be written when the I2C Slave is disabled.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		HSMEN	IGNACK	RXCFG	TXCFG	SAEN	GCEN	0			0	ACKSTALL	TXDSTALL	RXSTALL	ADRSTALL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved
23-19 —	Reserved
18-16 ADDRCFG	Address Configuration Configures the condition that causes an address to match. 000b - Address match 0 (7-bit) 001b - Address match 0 (10-bit) 010b - Address match 0 (7-bit) or Address match 1 (7-bit) 011b - Address match 0 (10-bit) or Address match 1 (10-bit) 100b - Address match 0 (7-bit) or Address match 1 (10-bit) 101b - Address match 0 (10-bit) or Address match 1 (7-bit) 110b - From Address match 0 (7-bit) to Address match 1 (7-bit) 111b - From Address match 0 (10-bit) to Address match 1 (10-bit)
15-14 —	Reserved
13 HSMEN	High Speed Mode Enable Enables detection of the High-Speed Mode master code of slave address 0000_1XX, but does not cause an address match on this code. When set and any HS-mode master code is detected, SCR[FILTEN] and SCFGR1[ACKSTALL] bits are ignored until the next STOP condition is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disables detection of HS-mode master code 1b - Enables detection of HS-mode master code
12 IGNACK	Ignore NACK When Ignore NACK is set, the LPI2C slave continues transfers after a NACK is detected. Ignore NACK bit is required to be set in Ultra-Fast mode. 0b - Slave ends transfer when NACK is detected 1b - Slave does not end transfer when NACK detected
11 RXCFG	Receive Data Configuration 0b - Reading the Receive Data register returns received data and clears the Receive Data flag (MSR[RDF]). 1b - Reading the Receive Data register when the Address Valid flag (SSR[AVF])is set, returns the Address Status register and clear the Address Valid flag. Reading the Receive Data register when the Address Valid flag is clear, returns received data and clears the Receive Data flag (MSR[RDF]).
10 TXCFG	Transmit Flag Configuration The transmit data flag always asserts before a NACK is detected at the end of a slave-transmit transfer. This can cause an extra word to be written to the transmit data FIFO. • When TXCFG = 0, the Transmit Data register is automatically emptied when a slave-transmit transfer is detected. This causes the transmit data flag to assert whenever a slave-transmit transfer is detected, and causes the transmit data flag to negate at the end of the slave-transmit transfer. • When TXCFG = 1, the Transmit Data flag asserts whenever the Transmit Data register is empty, and the Transmit Data flag negates when the Transmit Data register is full. This allows the Transmit Data register to be filled before a slave-transmit transfer is detected, but can cause the Transmit Data register to be written before a NACK is detected on the last byte of a slave transmit transfer. 0b - Transmit Data Flag only asserts during a slave-transmit transfer when the Transmit Data register is empty 1b - Transmit Data Flag asserts whenever the Transmit Data register is empty
9 SAEN	SMBus Alert Enable Enables a match on an SMBus alert. 0b - Disables match on SMBus Alert 1b - Enables match on SMBus Alert
8 GCEN	General Call Enable Enables a general call address. 0b - General Call address is disabled 1b - General Call address is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-5 —	Reserved
4 —	Reserved
3 ACKSTALL	ACK SCL Stall Enables SCL clock stretching during slave-transmit address byte(s) and slave-receiver address and data byte(s), to allow software to write the Slave Transmit ACK (STAR) register before the ACK or NACK is transmitted. Clock stretching occurs when transmitting the 9th bit, and is therefore not compatible with high speed mode. When ACKSTALL is enabled, there is no need to set either RX SCL Stall (SCFGR1[RXSTALL]) or Address SCL Stall (SCFGR1[ADRSTALL]). When ACKSTALL is enabled and there is an address match on the first byte of a 10-bit address, SSR[AVF] sets allowing software to read the Received Address before writing to the Slave Transmit ACK (STAR) register. 0b - Clock stretching is disabled 1b - Clock stretching is enabled
2 TXDSTALL	TX Data SCL Stall Enables SCL clock stretching when the SSR[TDF] = 1 during a slave-transmit transfer. Clock stretching occurs following the 9th bit, and is therefore compatible with high speed mode. 0b - Clock stretching is disabled 1b - Clock stretching is enabled
1 RXSTALL	RX SCL Stall Enables SCL clock stretching when SSR[RDF] = 1 during a slave-receive transfer. Clock stretching occurs following the 9th bit, and is therefore compatible with high speed mode. 0b - Clock stretching is disabled 1b - Clock stretching is enabled
0 ADRSTALL	Address SCL Stall Enables SCL clock stretching when SSR[AVF] = 1. Clock stretching only occurs following the 9th bit, and is therefore compatible with high speed mode. 0b - Clock stretching is disabled 1b - Clock stretching is enabled

68.5.1.24 Slave Configuration 2 (SCFGR2)

Offset

Register	Offset
SCFGR2	128h

Function

The Slave Configuration Register 2 should only be written when the I2C Slave is disabled.

Diagram

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

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

Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch Filter SDA Configures the I2C slave digital glitch filters for SDA input. • A configuration of 0 disables the glitch filter • Glitches equal to or less than FILTSDA cycles long are filtered out and ignored • The latency through the glitch filter is equal to FILTSDA+3 cycles, and must be configured to be less than the minimum SCL low or high period • The glitch filter cycle count is not affected by the PRESCALE configuration, and the glitch filter cycle count is disabled in high speed mode
23-20 —	Reserved
19-16 FILTSCL	Glitch Filter SCL Configures the I2C slave digital glitch filters for SCL input.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• A configuration of 0 disables the glitch filter • Glitches equal to or less than FILTSCL cycles long are filtered out and ignored • The latency through the glitch filter is equal to FILTSCL+3 cycles, and must be configured to be less than the minimum SCL low or high period • The glitch filter cycle count is not affected by the PRESCALE configuration, and the glitch filter cycle count is disabled in high speed mode
15-14 —	Reserved
13-8 DATAVD	Data Valid Delay Configures the SDA data valid delay time for the I2C slave, and is equal to FILTSCL+DATAVD+3 cycles. • The data valid delay must be configured to be less than the minimum SCL low period • The I2C slave data valid delay time is not affected by the PRESCALE configuration, and the I2C slave data valid delay time is disabled in high speed mode
7-4 —	Reserved
3-0 CLKHOLD	Clock Hold Time Configures the minimum clock hold time for the I2C slave, when clock stretching is enabled. • The minimum hold time is equal to CLKHOLD+3 cycles • The I2C slave clock hold time is not affected by the PRESCALE configuration, and the I2C slave clock hold time is disabled in high speed mode

68.5.1.25 Slave Address Match (SAMR)

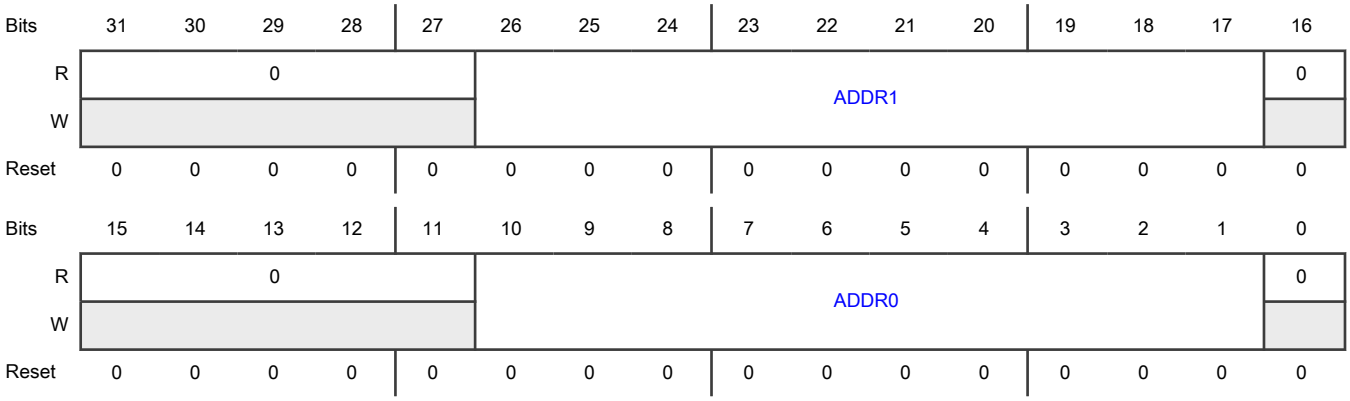
Offset

Register	Offset
SAMR	140h

Function

The SAMR should only be written when the I2C Slave is disabled.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-17 ADDR1	Address 1 Value Compared against the received address to detect the Slave Address. • In 10-bit mode, the first address byte is compared to { 11110, ADDR1[26:25] } and the second address byte is compared to ADDR1[24:17] • In 7-bit mode, the address is compared to ADDR1[23:17]
16-11 —	Reserved
10-1 ADDR0	Address 0 Value Compared against the received address to detect the Slave Address. • In 10-bit mode, the first address byte is compared to { 11110, ADDR0[10:9] } and the second address byte is compared to ADDR0[8:1] • In 7-bit mode, the address is compared to ADDR0[7:1] •
0 —	Reserved

68.5.1.26 Slave Address Status (SASR)

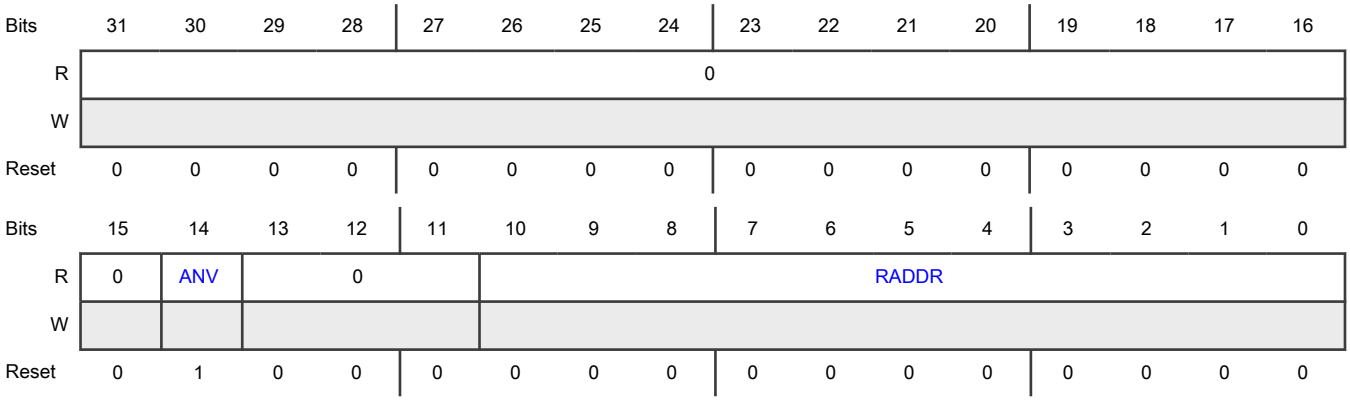
Offset

Register	Offset
SASR	150h

Function

SASR contains the read-only RADDR and ANV fields.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 ANV	Address Not Valid 0b - Received Address (RADDR) is valid 1b - Received Address (RADDR) is not valid
13-11 —	Reserved
10-0 RADDR	Received Address The Received Address updates whenever the AMF is set; the AMF is cleared by reading the Slave Address Status register. • In 7-bit mode, the address byte is stored in RADDR[7:0] • In 10-bit mode, the first address byte is { 11110, RADDR[10:9], RADDR[0] } and the second address byte is RADDR[8:1]. The R/W bit is therefore always stored in RADDR[0] • When ACKSTALL is set, if the first address byte matches in 10-bit mode, then the first address byte is stored in RADDR[7:0] so software can read the Received Address before writing the Transmit ACK. The Received Address then updates with the full 10-bit address if the second address byte matches.

68.5.1.27 Slave Transmit ACK (STAR)

Offset

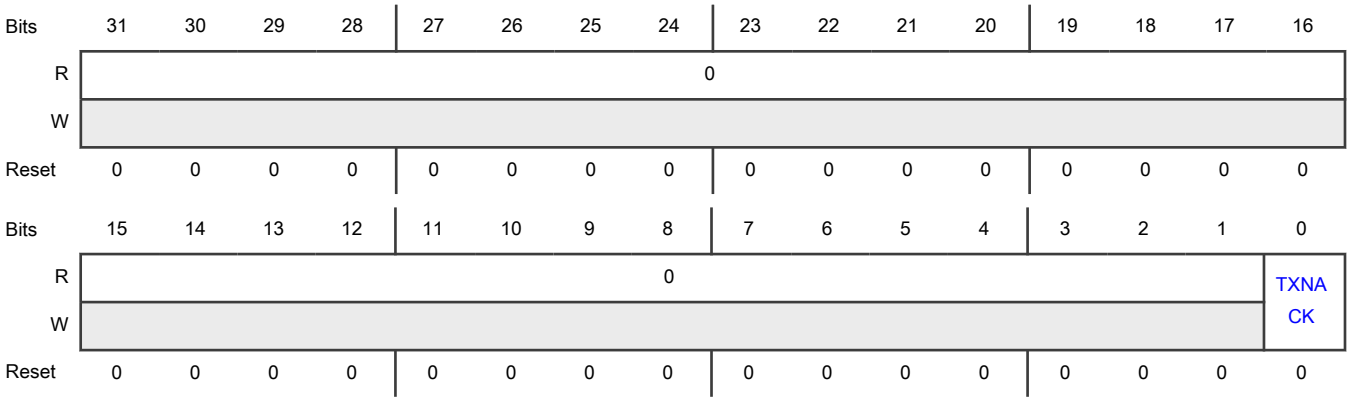
Register	Offset
STAR	154h

Function

STAR can only be written when the ACK SCL Stall bit is set (SCFGR1[ACKSTALL]).

- The SCFGR1[ACKSTALL] bit enables clock stretching during the ACK/NACK bit slot, and during this time, the STAR register can be written by software.
- The logic ensures that the clock stretching continues for at least 1 bus clock cycle after the STAR register is updated.
- This clock stretching time can be extended more using the Clock Hold Time field (SCFGR2[CLKHOLD]).

Diagram



Fields

Field	Function
31-1 —	Reserved
0 TXNACK	Transmit NACK After receiving each word, software can transmit either an ACK (logic 0) or a NACK (logic 1); Transmit NACK selects which to use: ACK or NACK. • When SCFGR1[ACKSTALL] is set, a Transmit NACK must be written once for each matching address byte and each received word. ACKSTALL must be set, because that stalls the data transfer until software reads the received word (and decides whether to respond with an ACK or NACK). • To configure the default ACK/NACK, Transmit NACK can also be written when LPI2C Slave is disabled or idle. 0b - Write a Transmit ACK for each received word 1b - Write a Transmit NACK for each received word

68.5.1.28 Slave Transmit Data (STDR)

Offset

Register	Offset
STDR	160h

Function

Clock stretching (enabled or disabled) affects when the transmit data is transferred. The SCFGR1[TXDSTALL] bit enables clock stretching during the 1st data bit of a slave-transmit transfer.

- **If clock stretching is enabled (SCFGR1[TXDSTALL] = 1)**, then the transmit data transfer is stalled until the STDR is updated. Clock stretching is extended by at least 1 bus clock cycle after STDR is updated, and clock stretching can be delayed even more using the Clock Hold Time field (SCFGR2[CLKHOLD].
- **If clock stretching is disabled (SCFGR1[TXDSTALL] = 0)**, then the transmit data should be written before the start of the slave-transmit transfer; otherwise (that is, if the transmit data is not written before the start of the slave-transmit transfer), the FIFO Error Flag (SSR[FEF]) sets.

Diagram

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

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

Fields

Field	Function
31-8 —	Reserved
7-0 DATA	Transmit Data Writing to the Slave Transmit Data Register (STDR) stores I2C slave transmit data *in* the Slave Transmit Data Register

68.5.1.29 Slave Receive Data (SRDR)

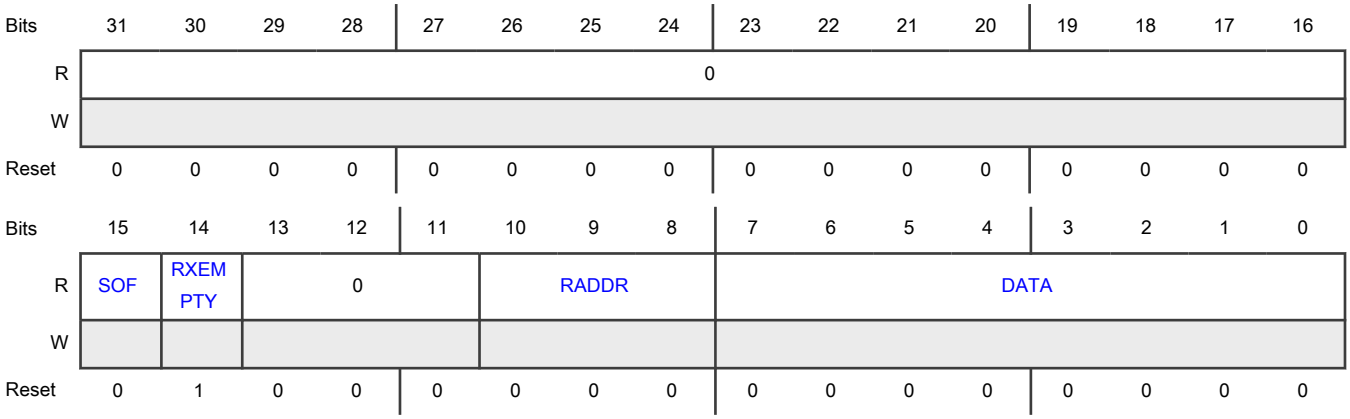
Offset

Register	Offset
SRDR	170h

Function

Reading the Slave Receive Data Register returns the data received by the I2C slave.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 SOF	Start Of Frame Indicates whether this is the first data word. 0b - Indicates this is not the first data word since a (repeated) START or STOP condition 1b - Indicates this is the first data word since a (repeated) START or STOP condition
14 RXEMPTY	RX Empty Gives the empty status of the receive data register. 0b - The Receive Data Register is not empty 1b - The Receive Data Register is empty
13-11 —	Reserved
10-8	Received Address

Table continues on the next page...

Table continued from the previous page...

Field	Function
RADDR	When both SCFGR1[RXCFG] and SSR[AVF] are set, the Address Status Register is returned. Otherwise this field will return zero.
7-0 DATA	Receive Data Contains data received by the I2C slave.

68.6 Glossary

HREQ	Host request
SCL	Serial clock line
SCLS	Secondary serial clock line
SDA	Serial data line
SDAS	Secondary serial data line

Chapter 69
Flexible I/O (FlexIO)

69.1 Chip-specific FlexIO information

69.1.1 FlexIO instances and configuration

The device contains one instance of FlexIO.

FlexIO is a highly configurable module providing a wide range of functionality including:

- Emulation of a variety of serial communication protocols like UART, I2C, SPI, SAI. It can work either as master or slave.
- Flexible 16-bit timers with support for a variety of trigger, reset, enable and disable conditions.

Table 390. FlexIO instances

Chip	Instance	No. of pins	No. of timers	No. of shift registers	No. of bits in counter
MWCT2016S, MWCT2D16S, MWCT2D17S,	FlexIO	32	8	8	16
MWCT2015S	FlexIO	16	8	8	16

Table 391. Data rate limitation

FlexIO operation	Data rate limitation
Master Tx	FlexIO_clk/4
Slave Tx	FlexIO_clk/10
Master Rx	FlexIO_clk/8
Slave Rx	FlexIO_clk/6

NOTE

FlexIO supports 32 bit accesses only.

NOTE

When I2S is emulated on FlexIO, you could observe a skew between bit-clock (BCLK) and frame sync pulse amounting to 1/2 of BCLK.

69.2 Overview

The FlexIO is a highly configurable module providing a wide range of functionality including:

- Emulation of a variety of serial/parallel communication protocols
- Flexible 16-bit timers with support for a variety of trigger, reset, enable and disable conditions
- Programmable logic blocks allowing the implementation of digital logic functions on-chip and configurable interaction of internal and external modules
- Programmable state machine for offloading basic system control functions from CPU

69.2.1 Block Diagram

The following diagram gives a high-level overview of the configuration of FlexIO timers and shifters.

The FlexIO uses shifters, timers and external triggers to shift data into or out of the FlexIO. The timing of the data shift is controlled by the timers, as shown in the block diagram. The timers can be configured to use generic timer functions, external triggers, or various other conditions to determine this control.

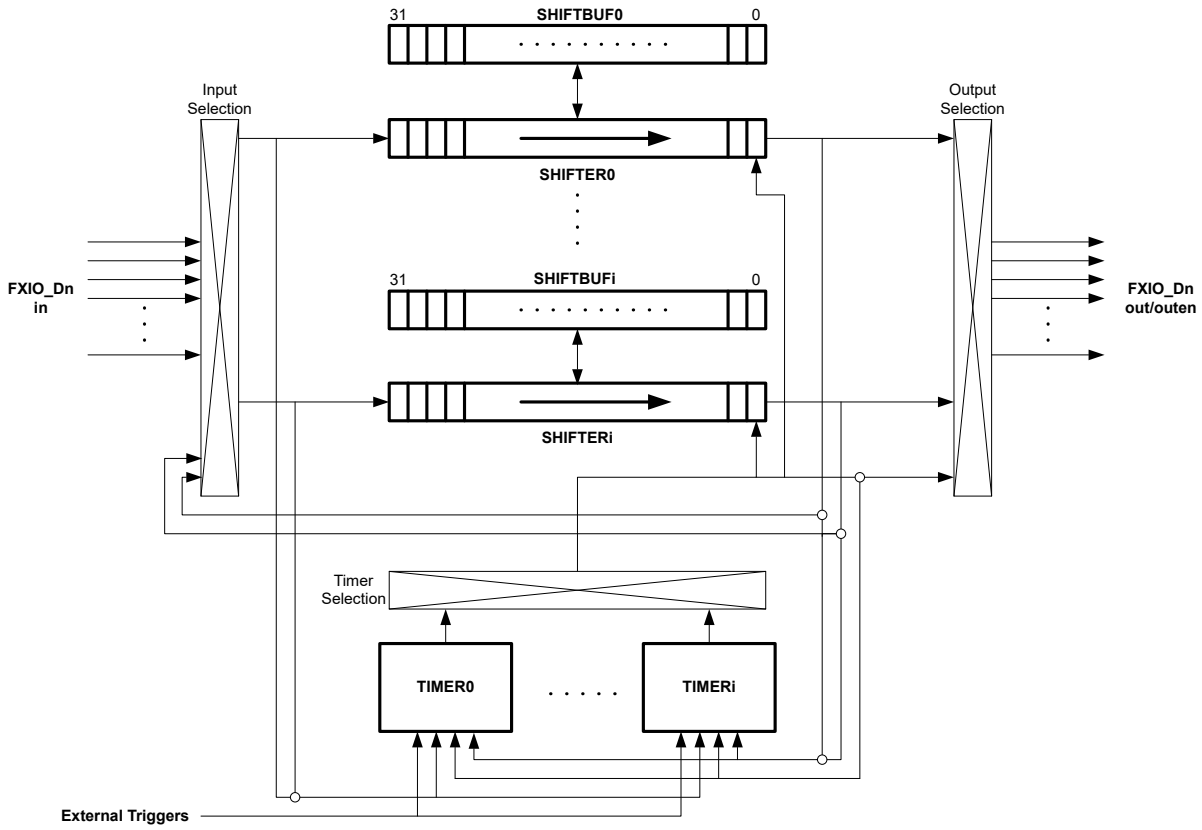


Figure 367. FlexIO block diagram

69.2.2 Features

The FlexIO module is capable of supporting a wide range of protocols including, but not limited to:

- UART
- I2C
- SPI
- I2S
- Camera IF
- Motorola 68K/Intel 8080 bus
- PWM/Waveform generation
- Input-capture (pulse edge interval measurement) - Example: SENT

The following key features are provided:

- Array of 32-bit shift registers with transmit, receive, data match, logic mode, and state modes
- Double buffered shifter operation for continuous data transfer

- Shifter concatenation to support large transfer sizes
- Automatic start/stop bit generation
- 1, 2, 4, 8, 16 or 32 multi-bit shift widths for parallel interface support
- Interrupt, DMA or polled transmit/receive operation
- Programmable baud rates independent of bus clock frequency
- Highly flexible 16-bit timers with support for a variety of internal or external trigger, reset, enable and disable conditions
- Programmable logic mode for integrating external digital logic functions on-chip or combining pin/shifter/timer functions to generate complex outputs
- Programmable state machine for offloading basic system control functions from CPU with support for up to 8 states, 8 outputs and 3 selectable inputs per state
- Integrated general purpose input/output registers and pin rising/falling edge interrupts to simplify software support

69.3 Functional description

69.3.1 Shifter operation

Shifters are responsible for buffering and shifting data into or out of the FlexIO. The timing of shift, load and store events are controlled by the Timer assigned to the Shifter via the SHIFTCTL[TIMSEL] register. The Shifters are designed to support either DMA, interrupt or polled operation. The following block diagram provides a detailed view of the Shifter microarchitecture.

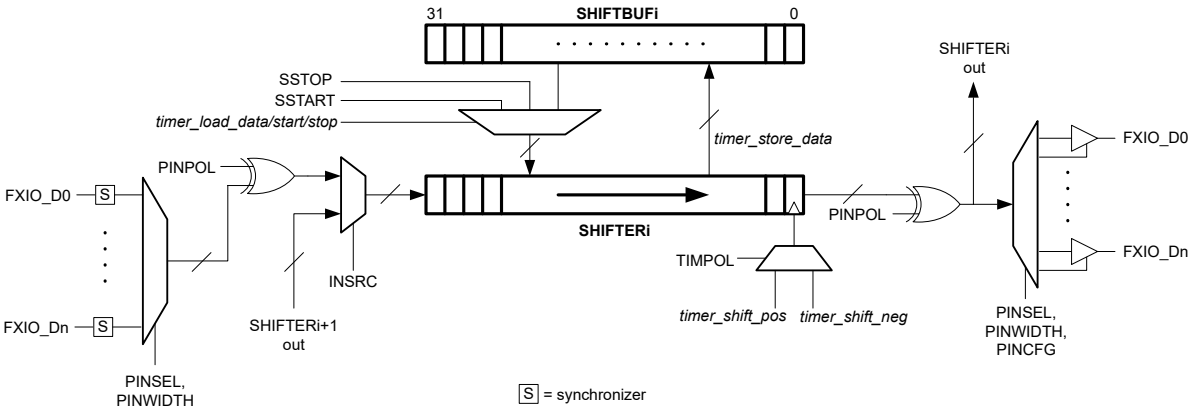


Figure 368. Shifter Microarchitecture

Transmit Mode

When configured for Transmit mode (SHIFTCTL[SMOD]=Transmit), the shifter loads data from the SHIFTBUF register and shift data out when a load event is signalled by the assigned Timer. An optional start/stop bit can also be automatically loaded before/ afterSHIFTBUFdata by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer.

NOTE

The shifter immediately loads a stop bit when the Shifter is initially configured for Transmit mode if a stop bit is enabled.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when data has been loaded from the SHIFTBUF register into the Shifter or when the Shifter is initially configured into Transmit mode. The flag clears when new data has been written into the SHIFTBUF register or a logic 1 is written to this flag. In transmit mode, any write of the SHIFTBUF register clears the corresponding Shifter Status Flag. It does not matter what is writing or the state of the DMA/interrupt enables, the flag is cleared. How the flag is set/cleared for each mode is documented in the SSF register description.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when an attempt to load data from an empty SHIFTBUF register occurs (buffer underrun). The flag can be cleared by writing it with logic 1.

Receive Mode

When configured for Receive mode (SHIFTCTL[SMOD]=Receive), the shifter shifts data in and store data into the SHIFTBUF register when a store event is signalled by the assigned Timer. Checking for a start/stop bit can be enabled before/after shifter data is sampled by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when data has been stored into the SHIFTBUF register from the Shifter. The flag clears when the data has been read from theSHIFTBUF register or a logic 1 is written to this flag. Any read of the SHIFTBUF register clears the corresponding Shifter Status Flag when Shifter is configured in Receive mode. It does not matter what is reading or the state of the DMA/interrupt enables, the flag is cleared. How the flag is set/clear for each mode is documented in the SSF register description.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when an attempt to store data into a fullSHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start/stop bit check. The flag can be cleared by writing it with logic 1.

Match Store Mode

When configured for Match Store mode (SHIFTCTL[SMOD]=Match Store), the shifter shifts data in, check for a match result and store matched data into the SHIFTBUF register when a store event is signalled by the assigned Timer. Checking for a start/stop bit can be enabled before/after shifter data is sampled by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer. Up to 16-bits of data can be compared using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when a match occurs and matched data has been stored into the SHIFTBUF register from the Shifter. The flag clears when the matched data has been read from the SHIFTBUF register or a logic 1 is written to this flag. Any read of the SHIFTBUF register clears the corresponding Shifter Status Flag when Shifter is configured in Match Store mode. It does not matter what is reading or the state of the DMA/interrupt enables, the flag is cleared. How the flag is set/clear for each mode is documented in the SSF register description.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when an attempt to store matched data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start/stop bit check. The flag can be cleared by writing it with logic 1.

Match Continuous Mode

When configured for Match Continuous mode (SHIFTCTL[SMOD]=Match Continuous), the shifter shifts data in and continuously check for a match result whenever a shift event is signalled by the assigned Timer. Up to 16-bits of data can be compared using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when a match occurs. The flag clears automatically as soon as there is no longer a match between Shifter data and SHIFTBUF register. **The flag cannot be cleared by reading the SHIFTBUF register.**

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when a match occurs. The flag clears when there is a read from the SHIFTBUF register or it written with logic 1.

State Mode

Using State mode enables the user to implement any state machine with up to 8 states, 8 outputs and 3 selectable inputs per state. This feature allows basic control functions to be offloaded from the CPU.

When configured for State mode (SHIFTCTL[SMOD]=State), the SHIFTBUF register is used to drive the output and compute next state values when the Shifter has been selected by the current state pointer (SHIFTSTATE[STATE]). The following diagram provides a detailed view of Shifter microarchitecture when configured for State mode.

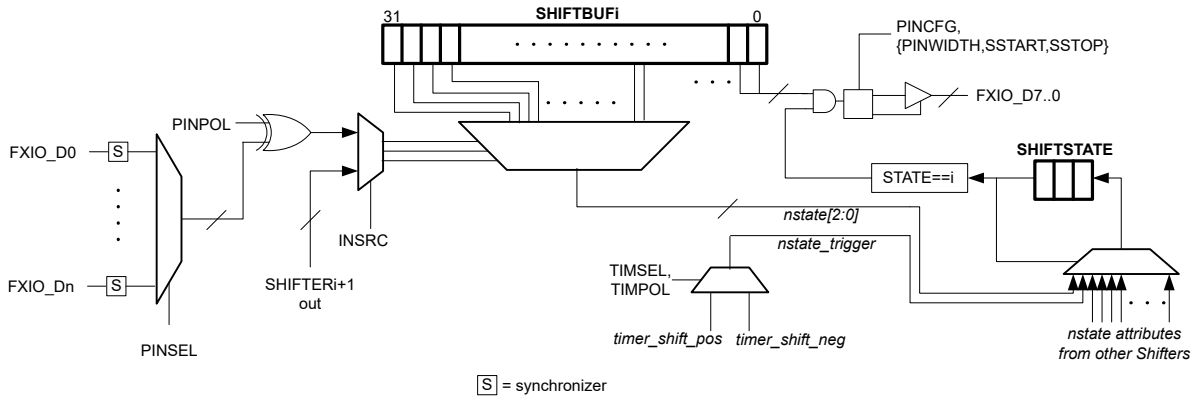


Figure 369. State Microarchitecture

When Shifter i has been selected by the current state pointer, output pins FXIO_D[7:0] are driven by SHIFTBUFi[31:24] using the configuration set by SHIFTCTLi[PINCFG]. When set, SHIFTCFGi{PWIDTH[3:0],SSTOP[1:0],SSTART[1:0]} are respectively used to disable the output drive on pins FXIO_D[7:0] for state machine applications which require less than 8 output pins.

The next state value is computed using the 3 input pins selected by SHIFTCTLi[PINSEL] together with SHIFTBUFi[23:0].

NOTE

Each state could potentially use a different set of 3 input pins.

The following table details how the next state value is computed when the current state pointer is pointing to Shifter i.

Table 392. Next State computation for SHIFTSTATE[STATE]=i

FXIO_D[PINSEL+2]	FXIO_D[PINSEL+1]	FXIO_D[PINSEL]	Next State Value
0	0	0	SHIFTBUFi[2:0]
0	0	1	SHIFTBUFi[5:3]
0	1	0	SHIFTBUFi[8:6]
0	1	1	SHIFTBUFi[11:9]
...	...	...	...
1	1	1	SHIFTBUFi[23:21]

Note that other shifters/timers could potentially be configured to drive the input pins of a given state, allowing the user to create complex combinations of shifters/timers as desired e.g. the output of a Shifter configured for logic mode could potentially be used to drive a state machine input.

The next state transition is triggered using the Timer output selected by SHIFTCTLi[TIMSEL] with polarity controlled by SHIFTCTLi[TIMPOL]. Note that each state could potentially use a different Timer to trigger each next state transition, allowing a variety of internal/external trigger sources and clocking configurations to be used (see Timer section for more detail).

The current state pointer defaults to Shifter 0 at reset, however it can be written by the user to select a different Shifter for the initial state. If the current state pointer selects a Shifter which is not configured for State mode, then outputs are not driven and a next state transition is never triggered.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set whenever the Shifter has been selected by the current state pointer. The flag clears when the current state pointer is updated to a different Shifter.

Logic Mode

Using logic mode enables the user to implement a small amount of programmable digital logic within a FlexIO Shifter.

When configured for Logic mode (SHIFTCTL[SMOD]=Logic), theSHIFTBUF register is used to implement a 5-input, 32-bit programmable logic look-up table. The following diagram provides a detailed view of Shifter microarchitecture when configured for Logic mode.

The SHIFTBUFi configures the lookup table for the 4 pin inputs and the SHIFTERi can optionally be used to configure a feedback or delayed pin source as the 5th input to the look up table.

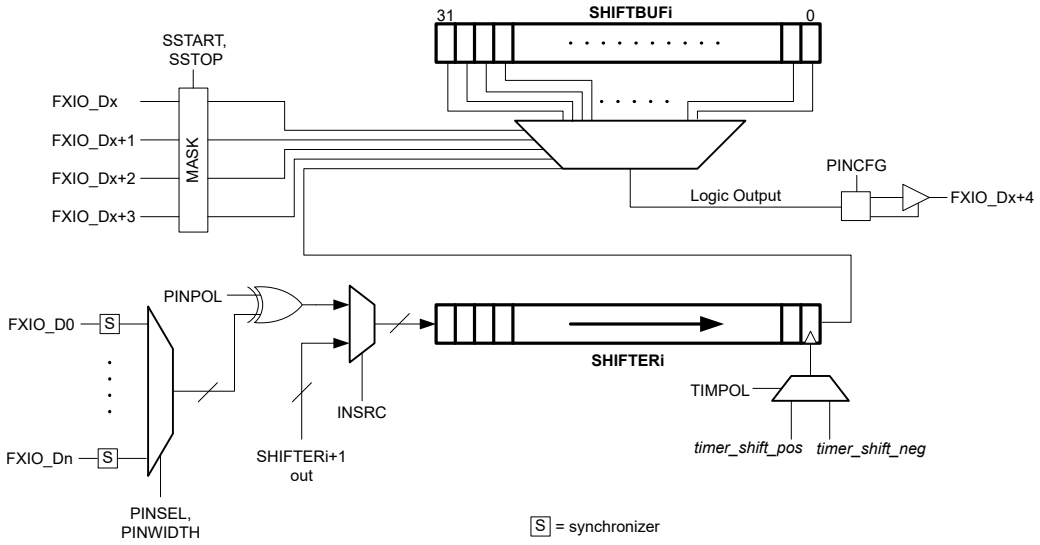


Figure 370. Logic Microarchitecture

The look-up table is driven using 4 pin inputs (maskable using SHIFTCFG[SSTOP] and SHIFTCFG[SSTART]) plus 1 input from the internal shifter and can be configured to drive an output pin using the SHIFTCTL[PINCFG] field. Pin inputs and outputs are fixed for each logic look-up table and are not selectable. The following table lists the logic output value selected by the look-up table for Shifter 'i'.

Table 393. Logic Look-up table for Shifter 'i'

SHIFTERi[0]	FXIO_D[x+3][1]	FXIO_D[x+2]	FXIO_D[x+1]	FXIO_D[x]	Logic Output to FXIO_D[x+4]
0	0	0	0	0	SHIFTBUFi[0]
0	0	0	0	1	SHIFTBUFi[1]
0	0	0	1	0	SHIFTBUFi[2]
0	0	0	1	1	SHIFTBUFi[3]
...	...	...	...	...	...
1	1	1	1	1	SHIFTBUFi[31]

1. for Shifter i=0...3, x=i

   for Shifter i=4...7, x=i+4

To minimize output glitches, SHIFTCFG[SSTOP] and SHIFTCFG[SSTART] can be used to mask unused input pins. When set, {SSTOP[1:0], SSTART[1:0]} mask FXIO_D[x+3]...FXIO_D[x] inputs respectively, so that any transitions on these pins does not cause the logic output to glitch.

Note that other shifters/timers could potentially be configured to drive the input pins of a given look-up table, allowing the user to concatenate look-up tables or create complex combinations of shifters/timers as desired.

SHIFTCFG[PWIDTH] controls the number of delay stages introduced by the internal shifter input (SHIFTERi[0]). For example, when configured for 1-bit shift (PWIDTH=0), the internal shifter introduces a 32 Shift clock delay before passing its input (selected by SHIFTCTL[PINSEL]) to the look-up table. When configured for 32-bit shift (PWIDTH=16...31), the internal shifter introduces a 1 Shift clock delay to its input.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set whenever the output pin allocated to the logic look-up table has a value of 1 (after being synchronized to the FlexIO clock). The flag clears when the output pin has a value of 0. This also allows the SSF flag to be used as a trigger to a Timer if desired.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when the output pin allocated to the logic look-up table has asserted. The flag can be cleared by writing it with logic 1.

The logic mode input pins (including pins driven by other shifters/timers) are first synchronized to the FlexIO functional clock before they are input to the programmable logic look-up table.

69.3.2 Timer Operation

The FlexIO 16-bit timers control the loading, shifting and storing of the shift registers, the counters load the contents of the compare register and decrement down to zero on the FlexIO clock. They can perform generic timer functions such as generating a clock or select output or a PWM waveform. Timers can be configured to enable in response to a trigger, pin or shifter condition; decrement always or only on a trigger or pin edge; reset in response to a trigger or pin condition; and disable on a trigger or pin condition or on a timer compare. Timers can optionally include a start condition and/or stop condition.

Each timer operates independently, although a timer can be configured to enable or disable at the same time as the previous timer (eg: timer1 can enable or disable at the same time as timer 0) and a timer output can be used to trigger any other timer. The trigger used by each timer is configured independently and can be configured to be a timer output, shifter status flag, pin input or an external trigger input (refer to the chip-specific FlexIO information for details on the external trigger connections). The trigger configuration is separate from the pin configuration, which can be configured for input, output data or output enable.

The Timer Configuration Register (TIMCFGn) should be configured before setting the Timer Mode (TIMOD).

Timer 8-bit Baud Counter Mode

In 8-bit Baud Counter Mode, the 16-bit counter is divided into two 8-bit counters. The lower 8-bits are used to configure the baud rate of the shift clock and the upper 8-bits are used to configure the number of shift clock edges in the transfer. When the lower 8-bits decrement to zero, the timer output is toggled and the lower 8-bits reload from the compare register. The upper 8-bits decrement when the lower 8-bits equal zero and decrement.

Note that a timer reset event in 8-bit Baud Counter Mode only resets the lower 8-bit counter, the upper 8-bit counter is not affected and can decrement if the timer reset is configured to update the state of the timer output, and the timer output toggles as a result of the timer reset event.

A timer compare event occurs when the upper 8-bits equal zero and decrements. The timer status flag is set on a timer compare event.

Timer 8-bit High PWM Mode

In 8-bit High PWM Mode, the 16-bit counter is divided into two 8-bit counters. The lower 8-bits are used to configure the timer output high period and the upper 8-bits are used to configure the timer output low period. The lower 8-bits decrement when the output is high. When the lower 8-bits equal zero and decrement, the timer output is cleared and the lower 8-bits are reloaded from the compare register. The upper 8-bits decrement when the output is low. When the upper 8-bits equal zero and decrement, the timer output is set and the upper 8-bits are reloaded from the compare register.

A timer compare event occurs when the upper 8-bits equal zero and decrements. The timer status flag is set on a timer compare event.

Timer 16-bit Counter Mode

In 16-bit Counter Mode, the 16-bit counter can be used to configure either the baud rate of the shift clock (eg: TIMDEC[1:0] != 10 or 11) or the number of shift clock edges in the transfer (eg: TIMDEC[1:0] = 10 or 11). When the 16-bit counter equals zero and decrements, the timer output toggles and the counter reloads from the compare register.

A timer compare event occurs when the 16-bit counter equals zero and decrements. The timer status flag is set on a timer compare event.

Timer 16-bit Counter Disable Mode

In 16-bit Counter Disable Mode, the 16-bit counter can be used to configure either the baud rate of the shift clock (eg: TIMDEC[1:0] != 10 or 11) or the number of shift clock edges in the transfer (eg: TIMDEC[1:0] = 10 or 11). When the 16-bit counter equals zero and decrements, the timer output toggles and the counter reloads from the compare register.

A timer compare event occurs when the 16-bit counter equals zero and decrements. The timer status flag is set on a timer disable event.

Timer 8-bit Word Counter Mode

In 8-bit Word Counter Mode, the 16-bit counter is divided into two 8-bit counters. The lower 8-bits are used to configure the number of shift clock edges in each word and the upper 8-bits are used to configure the number number of words in the transfer. When the lower 8-bits decrement to zero, the timer output is toggled and the lower 8-bits reload from the compare register. The upper 8-bits only decrement when the lower 8-bits equal zero and decrement.

A timer compare event occurs when the lower 8-bits equal zero and decrements. The timer status flag is set when the upper 8-bits equal zero and decrements.

Timer 8-bit Low PWM Mode

In 8-bit Low PWM Mode, the 16-bit counter is divided into two 8-bit counters. The lower 8-bits are used to configure the timer output low period and the upper 8-bits are used to configure the timer output high period. The lower 8-bits decrement when the output is low. When the lower 8-bits equal zero and decrement, the timer output is set and the lower 8-bits are reloaded from the compare register. The upper 8-bits decrement when the output is high. When the upper 8-bits equal zero and decrement, the timer output is cleared and the upper 8-bits are reloaded from the compare register.

A timer compare event occurs when the upper 8-bits equal zero and decrements. The timer status flag is set on a timer compare event.

Timer 16-bit Input Capture Mode

In 16-bit Input Capture Mode, the 16-bit counter uses a fixed initial value of 0xFFFF to load the counter. Whenever a timer counter disable condition is detected (as configured by TIMDIS), the timer status flag is set and the inverted timer counter value is stored in the timer compare register, then the timer counter is immediately restarted from 0xFFFF. The timer output is always output on the FlexIO trigger outputs (which can also be used by other timers as an input trigger) and can also be output to a pin (PINCFG/PINPOL/PINSEL) or a shifter.

If the timer status flag is already set, then the input capture event is missed. When the 16-bit counter equals zero and decrements, the timer output toggles and the counter reloads with 0xFFFF. The timer output can be used to count the number of counter overflows between two input capture events. Input capture mode is not intended to be used to control shift registers, it is to enable software to count the number of bits in a variable length transfer or to monitor the period or frequency of an input.

A timer compare event occurs when the 16-bit counter equals zero and decrements. The timer status flag is set on a timer disable event. Whenever the status flag asserts, the inverted timer counter value is stored in the timer compare register.

Timer Enable and Start Bit

When the TIMOD is configured for the desired mode, and the condition configured by timer enable (TIMENA) is detected then the following events occur. When ONETIM is set, then the timer status flag must also be clear to generate a timer enable event, otherwise the timer enable event is blocked. This can be used to enforce software intervention after each timer iteration.

- Timer counter loads the current value of the compare Register and start decrementing as configured by TIMDEC.
- Timer output may update to its initial state depending on the TIMOUT configuration. Shifters that are controlled by this timer do not see this as a rising edge on the timer shift clock.
- Transmit shifters controlled by this timer either output their start bit value, or load the shift register from the shift buffer and output the first bit, as configured by SSTART.

If the Timer start bit is enabled, the timer counter reloads with the compare register on the first rising edge of the shift clock after the timer starts decrementing. If there is no falling edge on the shift clock before the first rising edge (for example, when TIMOUT=1), a shifter that is configured to shift on falling edge and load on the first shift does not load correctly.

Timer Decrement and Reset

The Timer then generates the timer output and timer shift clock depending on the TIMOD and TIMDEC fields. The shifter clock is either equal to the timer output (when TIMDEC != 10 or 11) or equal to the decrement clock (when TIMDEC = 10 or 11). When TIMDEC is configured to decrement from a pin or trigger, the timer decrements on both rising and falling edges.

If a Timer is configured to decrement on the FlexIO functional clock divided by 16 or 256 (when TIMDEC = 100 or 101), then a common prescaler that is shared by all Timers is used to generate the two divide ratios. This prescaler is reset when all Timers are either idle or configured to not use the prescaler (TIMDEC != 100 or 101).

When the Timer is configured to reset as configured in the TIMRST field then the Timer counter loads the current value of the compare register again. The timer output and timer shift clock can be configured to update on timer reset, as configured by TIMOUT. This can result in a timer shift clock edge if the timer output toggles as a result of the timer reset. In 8-bit Baud Counter mode this would also decrement the upper 8-bits of the counter.

In general, when the timer counter decrements to zero a timer compare event is triggered. The timer compare event causes the timer counter to load the contents of the timer compare register, the timer output to toggle, any configured transmit shift registers to load, and any configured receive shift registers to store. Depending on the timer mode, the timer status flag may also be set.

Timer Disable and Stop Bit

When the is Timer is configured to add a stop bit on each compare, the following additional events occur.

- Transmit shifters controlled by this timer output their stop bit value (if configured by SSTOP).
- Receive shifters controlled by this timer store the contents of the shift register in their shift buffer, as configured by SSTOP.
- On the first rising edge of the shifter clock after the compare, the timer counter reloads the current value of the Compare Register.

Transmit shifters must be configured to load on the first shift when the timer is configured to insert a stop bit on each compare.

When the condition configured by timer disable (TIMDIS) is detected, the following events occur.

- Timer counter reloads the current value of the Compare Register and start decrementing as configured by TIMDEC.
- Timer output clears. Shifters that are controlled by this timer do not see this as a falling edge on the timer shift clock, but can generate a shift event if the timer shift clock would otherwise generate one.
- Transmit shifters controlled by this timer output their stop bit value (if configured by SSTOP).
- Receive shifters controlled by this timer store the contents of the shift register in their shift buffer, as configured by SSTOP.

If the timer stop bit is enabled, the timer counter continues decrementing until the next rising edge of the shift clock is detected, at which point it finishes. Although the timer output is forced low during the stop bit, the timer shift clock can toggle during the stop bit, but does not generate shift events.

A timer enable condition can be detected in the same cycle as a timer disable condition (if timer stop bit is disabled), or on the first rising edge of the shift clock after the disable condition (if stop bit is enabled). When ONETIM is set, the timer status flag must be clear before the next timer enable condition is detected. Receive shift registers with stop bit enabled store the contents of the shift register into the shift buffer and verify the state of the input data on the configured shift edge while the timer is in the stop state

condition. If there is no configured edge between the timer disable and the next rising edge of the shift clock then the final store and verify do not occur.

69.3.3 Pin operation

The pin configuration for each timer and shifter can be configured to use any FlexIO pin with either polarity. Each timer and shifter can be configured as an input, output data, output enable or bidirectional output. A pin configured for output enable can be used as an open drain (with inverted polarity, since the output enable assertion would cause logic zero to be output on the pin) or to control the enable on the bidirectional output. Any timer or shifter could be configured to control the output enable for a pin where the bidirectional output data is driven by another timer or shifter.

Parallel Interface

Shifters can be configured to use multiple FlexIO pins in parallel using the SHIFTCFG[PWIDTH] field. PWIDTH is used to configure the following settings of a shifter:

1. Number of bits shifted per Shift clock.
2. Number of pins driven by the shifter per Shift clock (only on shifters supporting parallel transmit i.e. SHIFTER0, SHIFTER4).
3. Number of pins sampled by the shifter per Shift clock (only on Shifter supporting parallel receive i.e. SHIFTER3, SHIFTER7).

When configured for parallel shift, either 4, 8, 16 or 32-bits can be shifted on every Shift clock. If an adjacent shifter is selected as the input source (SHIFTCFG[INSRC]=1), the least significant 4, 8, 16 or 32-bits from the adjacent shifter is sampled on each Shift clock.

For shifters supporting parallel receive (SHIFTER3, SHIFTER7), the shifter can be configured to sample multiple pins (SHIFTCFG[INSRC]=0), with PWIDTH and PINSEL selecting the pins as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL]. Note that if PWIDTH is less than the number of bits being shifted on each Shift clock, then the most significant bits are masked with 0 e.g. if PINSEL=7 and PWIDTH=6, then SHIFTER[31:24] samples {0,0,FXIO_D[12:7]} on each Shift clock.

For shifters supporting parallel transmit (SHIFTER0, SHIFTER4), the shifter can be configured to drive multiple pins using SHIFTCTL[PINCFG], with PWIDTH and PINSEL selecting the pins as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL]. Note that if PWIDTH is less than the number of bits being shifted on each Shift clock, then the most significant pins are not driven e.g. if PINSEL=7 and PWIDTH=6, then SHIFTER[5:0] drives only FXIO_D[12:7] on each Shift clock.

Pin Synchronization

When configuring a pin as an input (this includes a timer trigger configured as a pin input), the input signal is first synchronized to the FlexIO clock before the signal is used by a timer or shifter. This introduces a small latency of between 0.5 to 1.5 FlexIO clock cycles when using an external pin input to generate an output or control a shifter. This sets the maximum setup time at 1.5 FlexIO clock cycles.

If an input is used by more than one timer or shifter then the synchronization occurs once to ensure any edge is seen on the same cycle by all timers and shifters using that input.

Note that FlexIO pins are also connected internally, configuring a FlexIO shifter or timer to output data on an unused pin makes an internal connection that allows other shifters and timer to use this pin as an input. This allows a shifter output to be used to trigger a timer or a timer output to be shifted into a shifter. This path is also synchronized to the FlexIO clock and therefore incurs a 1 cycle latency.

So when using a Pin input as a Timer Trigger, Timer Clock or Shifter Data Input, the following synchronization delays occur:

1. 0.5 to 1.5 FlexIO clock cycles for external pin
2. 1 FlexIO clock cycle for an internally driven pin

For timing considerations such as output valid time and input setup time for specific applications (SPI Master, SPI Slave, I2C Master, I2S Master, I2S Slave) please refer to the FlexIO Application Information Section.

Pin Override

The state of any FlexIO pin can be overriden at any time by software. The Pin Output Enable register configures any pin as an output and drives that pin with the value in the Pin Output Data Register.

Alias registers for the Pin Output Enable/Data Registers also exist, writing a logic one to an alias register updates the corresponding register bits in both the Pin Output Enable Register and the Pin Output Data Register as follows.

- Pin Output Disable Register clears Output Enable Register and clear Output Data Register.
- Pin Output Clear Register sets Output Enable Register and clear Output Data Register.
- Pin Output Set Register sets Output Enable Register and set Output Data Register.
- Pin Output Toggle Register sets Output Enable Register and toggle Output Data Register.

Pin Interrupt

The state of any FlexIO pin can be read by software at any time. Software can also configure any pin to set a status flag when either a rising and/or falling edge is detected on that pin. The pin status flag can also be configured to generate an interrupt.

69.3.4 Low Power Modes

The FlexIO remains functional during low power modes, provided the FlexIO functional clock remains enabled.

69.3.5 Debug Mode

The FlexIO remains functional in debug mode provided the Debug Enable bit (CTRL[DBGE]) is set.

69.3.6 Clocks

Functional clock

The FlexIO functional clock is asynchronous to the bus clock and can remain enabled in low power modes. The FlexIO functional clock must be enabled before accessing any FlexIO registers. Provided the FlexIO functional clock is at least equal to the bus clock, the CTRL[FASTACC] bit can be set to support fast register accesses.

Bus clock

The bus clock is only used for bus accesses to the control and configuration registers.

69.3.7 Reset

Chip reset

The logic and registers for the FlexIO are reset to their default state on a chip reset.

Software reset

The FlexIO implements a software reset bit in its Control Register. The CTRL[SWRST] resets all logic and registers to their default state, except for the CTRL itself.

69.3.8 Interrupts and DMA Requests

The following table illustrates the status flags that can generate the FlexIO interrupt and DMA requests.

Table 394. FlexIO Interrupts and DMA Requests

Flag	Description	Interrupt	DMA Request	Low Power Wakeup
SSF	Shifter Status Flag.	Y	Y	Y
SEF	Shifter Error Flag.	Y	N	Y
TSF	Timer Status Flag.	Y	Y	Y
PSF	Pin Status Flag.	Y	N	Y
ETSF	External Trigger Status Flag.	Y	N	Y

69.3.9 Peripheral Triggers

The connection of the FlexIO peripheral triggers with other peripherals are device specific.

Output Triggers

Each FlexIO Timer generates an output trigger equal to the timer output. The output trigger is not affected by the timer pin polarity configuration.

Input Trigger

FlexIO supports multiple external trigger inputs that can be used to trigger one or more FlexIO timers. If a rising edge is detected on an external trigger when the FlexIO is enabled, then the external trigger status flag is set. The external triggers are synchronized to the FlexIO functional clock and must assert for at least two cycles of the FlexIO functional clock to be sampled correctly.

69.4 Application Information

This section provides examples for a variety of FlexIO module applications.

69.4.1 UART Transmit

UART transmit can be supported using one Timer, one Shifter and one Pin (two Pins if supporting CTS). The start and stop bit insertion is handled automatically and multiple transfers can be supported using DMA controller. The timer status flag can be used to indicate when the stop bit of each word is transmitted.

Break and idle characters require software intervention, before transmitting a break or idle character the SSTART and SSTOP fields should be altered to transmit the required state and the data to transmit must equal 0xFF or 0x00. Supporting a second stop bit requires the stop bit to be inserted into the data stream using software (and increasing the number of bits to transmit). Note that when performing byte writes to SHIFTBUFn (or SHIFTBUFBIS for transmitting MSB first), the rest of the register remains unaltered allowing an address mark bit or additional stop bit to remain undisturbed.

FlexIO does not support automatic insertion of parity bits.

Table 395. UART Transmit Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0. Can invert output data by setting PINPOL, or can support open drain by setting PINPOL=0x1 and PINCFG=0x1.

Table continues on the next page...

Table 395. UART Transmit Configuration (continued)

Register	Value	Comments
TIMCMPn	0x0000_0F01	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0000_2222	Configure start bit, stop bit, enable on trigger asserted and disable on compare. Can support CTS by configuring TIMEN=0x3.
TIMCTLn	0x01C0_0001	Configure dual 8-bit counter using Shifter 0 status flag as inverted internal trigger source. Can support CTS by configuring PINSEL=0x1 (for Pin 1) and PINPOL=0x1.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit transfer, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS[7:0] register instead.

An alternative configuration is possible that supports slower baud rates but requires two Timers.

Table 396. UART Transmit Configuration for Slow Baud Rate

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0. Can invert output data by setting PINPOL, or can support open drain by setting PINPOL=0x1 and PINCFG=0x1.
TIMCMPn	0x0000_000F	Configure for 8-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFGn	0x0030_2622	Configure start bit, stop bit, enable on trigger rising edge, decrement on trigger and disable on compare.
TIMCTLn	0x0740_0003	Configure 16-bit counter using Timer 1 output as internal trigger source.
TIMCMP(n+1)	0x0000_0001	Configure baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:0] = (baud rate divider / 2) - 1.

Table continues on the next page...

Table 396. UART Transmit Configuration for Slow Baud Rate (continued)

Register	Value	Comments
TIMCFG(n+1)	0x0000_1200	Configure enable on trigger asserted and disable on Timer 0 disable. Can support CTS by configuring TIMEN=0x3.
TIMCTL(n+1)	0x01C0_0003	Configure 16-bit counter using Shifter 0 status flag as inverted internal trigger source. Can support CTS by configuring PINSEL=0x1 (for Pin 1) and PINPOL=0x1.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit transfer, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS[7:0] register instead.

69.4.2 UART Receive

UART receive can be supported using one Timer, one Shifter and one Pin (two Timers and two Pins if supporting RTS). The start and stop bit verification is handled automatically and multiple transfers can be supported using the DMA controller. The timer status flag can be used to indicate when the stop bit of each word is received.

Triple voting of the received data is not supported by FlexIO, data is sampled only once in the middle of each bit. Another timer can be used to implement a glitch filter on the incoming data, another Timer can also be used to detect an idle line of programmable length. Break characters cause the error flag to set and the shifter buffer register returns 0x00.

FlexIO does not support automatic verification of parity bits.

Table 397. UART Receiver Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0080_0001	Configure receive using Timer 0 on negedge of clock with input data on Pin 0. Can invert input data by setting PINPOL.
TIMCMPn	0x0000_0F01	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0204_2422	Configure start bit, stop bit, enable on pin posedge and disable on compare. Enable resynchronization to received data with TIMOUT=0x2 and TIMRST=0x4.

Table continues on the next page...

Table 397. UART Receiver Configuration (continued)

Register	Value	Comments
TIMCTLn	0x0000_0081	Configure dual 8-bit counter using inverted Pin 0 input.
SHIFTBUFn	Data to receive	Received data can be read from SHIFTBUFBYS[7:0], use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS[7:0] register instead.

The UART Receiver with RTS configuration uses a 2nd Timer to generate the RTS output. The RTS asserts when the start bit is detected and negate when the data is read from the shifter buffer register. No start bit is detected while the RTS is asserted, the received data is simply ignored.

Table 398. UART Receiver with RTS Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0080_0001	Configure receive using Timer 0 on negedge of clock with input data on Pin 0. Can invert input data by setting PINPOL.
TIMCMPn	0x0000_0F01	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0204_2522	Configure start bit, stop bit, enable on pin posedge with trigger asserted and disable on compare. Enable resynchronization to received data with TIMOUT=0x2 and TIMRST=0x4.
TIMCTLn	0x02C0_0081	Configure dual 8-bit counter using inverted Pin 0 input. Trigger is internal using inverted Pin 1 input.
TIMCMP(n+1)	0x0000_FFFF	Never compare.
TIMCFG(n+1)	0x0030_6100	Enable on Timer N enable and disable on trigger falling edge. Decrement on trigger to ensure no compare.
TIMCTL(n+1)	0x0143_0003	Configure 16-bit counter and output on Pin 1. Trigger is internal using Shifter 0 flag.
SHIFTBUFn	Data to receive	Received data can be read from SHIFTBUFBYS[7:0], use the Shifter Status Flag to indicate when data

Table continues on the next page...

Table 398. UART Receiver with RTS Configuration (continued)

Register	Value	Comments
		can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS[7:0] register instead.

69.4.3 SPI Master

SPI master mode can be supported using two Timers, two Shifters and four Pins. Either CPHA=0 or CPHA=1 can be supported and transfers can be supported using the DMA controller. For CPHA=1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The stop bit is used to guarantee a minimum of 1 clock cycle between the slave select negating and before the next transfer. Writing to the transmit buffer by either core or DMA is used to initiate each transfer.

NOTE

Due to synchronization delays, the setup time for the serial input data is 1.5 FlexIO clock cycles, so the maximum baud rate is divide by 4 of the FlexIO clock frequency.

Table 399. SPI Master (CPHA=0) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0000	Start and stop bit disabled.
SHIFTCTLn	0x0083_0002	Configure transmit using Timer 0 on negedge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0000_0101	Configure receive using Timer 0 on posedge of clock with input data on Pin 1.
TIMCMPn	0x0000_3F01	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0100_2222	Configure start bit, stop bit, enable on trigger high and disable on compare, initial clock state is logic 0.
TIMCTLn	0x01C3_0201	Configure dual 8-bit counter using Pin 2 output (shift clock), with Shifter 0 flag as the inverted trigger. Set PINPOL to invert the output shift clock.
TIMCMP(n+1)	0x0000_FFFF	Never compare.
TIMCFG(n+1)	0x0000_1100	Enable when Timer 0 is enabled and disable when Timer 0 is disabled.

Table continues on the next page...

Table 399. SPI Master (CPHA=0) Configuration (continued)

Register	Value	Comments
TIMCTL(n+1)	0x0003_0383	Configure 16-bit counter (never compare) using inverted Pin 3 output (as slave select).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

Table 400. SPI Master (CPHA=1) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0021	Start bit loads data on first shift.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0080_0101	Configure receive using Timer 0 on negedge of clock with input data on Pin 1.
TIMCMPn	0x0000_3F01	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0100_2222	Configure start bit, stop bit, enable on trigger high and disable on compare, initial clock state is logic 0.
TIMCTLn	0x01C3_0201	Configure dual 8-bit counter using Pin 2 output (shift clock), with Shifter 0 flag as the inverted trigger. Set PINPOL to invert the output shift clock. Set TIMDIS=3 to keep slave select asserted for as long as there is data in the transmit buffer.
TIMCMP(n+1)	0x0000_FFFF	Never compare.

Table continues on the next page...

Table 400. SPI Master (CPHA=1) Configuration (continued)

Register	Value	Comments
TIMCFG(n+1)	0x0000_1100	Enable when Timer 0 is enabled and disable when Timer 0 is disabled.
TIMCTL(n+1)	0x0003_0383	Configure 16-bit counter (never compare) using inverted Pin 3 output (as slave select).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

69.4.4 SPI Slave

SPI slave mode can be supported using one Timer, two Shifters and four Pins. Either CPHA=0 or CPHA=1 can be supported and transfers can be supported using the DMA controller. For CPHA=1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The transmit data must be written to the transmit buffer register before the external slave select asserts, otherwise the shifter error flag is set.

NOTE

Due to synchronization delays, the output valid time for the serial output data is 2.5 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

Table 401. SPI Slave (CPHA=0) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0000	Start and stop bit disabled.
SHIFTCTLn	0x0083_0002	Configure transmit using Timer 0 on falling edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0000_0101	Configure receive using Timer 0 on rising edge of shift clock with input data on Pin 1.
TIMCMPn	0x0000_003F	Configure 32-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.

Table continues on the next page...

Table 401. SPI Slave (CPHA=0) Configuration (continued)

Register	Value	Comments
TIMCFGn	0x0120_0600	Configure enable on trigger rising edge, initial clock state is logic 0 and decrement on pin input.
TIMCTLn	0x06C0_0203	Configure 16-bit counter using Pin 2 input (shift clock), with Pin 3 input (slave select) as the inverted trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

Table 402. SPI Slave (CPHA=1) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0001	Shifter configured to load on first shift and stop bit disabled.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on rising edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0080_0101	Configure receive using Timer 0 on falling edge of shift clock with input data on Pin 1.
TIMCMPn	0x0000_003F	Configure 32-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFGn	0x0120_6602	Configure start bit, enable on trigger rising edge, disable on trigger falling edge, initial clock state is logic 0 and decrement on pin input.
TIMCTLn	0x06C0_0203	Configure 16-bit counter using Pin 2 input (shift clock), with Pin 3 input (slave select) as the inverted trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written

Table continues on the next page...

Table 402. SPI Slave (CPHA=1) Configuration (continued)

Register	Value	Comments
		using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

69.4.5 I2C Master

I2C master mode can be supported using two Timers, two Shifters and two Pins. One timer is used to generate the SCL output and one timer is used to control the shifters. The two shifters are used to transmit and receive for every word, when receiving the transmitter must transmit 0xFF to tristate the output. FlexIO inserts a stop bit after every word to generate/verify the ACK/NACK. FlexIO waits for the first write to the transmit data buffer before enabling SCL generation. Data transfers can be supported using the DMA controller and the shifter error flag sets on transmit underrun or receive overflow.

The first timer generates the bit clock for the entire packet (START to Repeated START/STOP), so the compare register needs to be programmed with the total number of clock edges in the packet (minus one). The timer supports clock stretching using the reset counter when pin equal to output (although this increases both the clock high and clock low periods by at least 1 FlexIO clock cycle each). The second timer uses the SCL input pin to control the transmit/receive shift registers, this enforces an SDA data hold time by an extra 2 FlexIO clock cycles.

Both the transmit and receive shifters need to be serviced for each word in the transfer, the transmit shifter must transmit 0xFF when receiving and the receive shifter returns the data actually present on the SDA pin. The transmit shifter loads 1 additional word on the last falling edge of SCL pin, this word should be 0x00 if generating a STOP condition or 0xFF if generating a repeated START condition. During the last word of a master-receiver transfer, the transmit SSTOP bit should be set by software to generate a NACK.

The receive shift register asserts an error interrupt if a NACK is detected, but software is responsible for generating the STOP or repeated START condition. If a NACK is detected during master-transmit, the interrupt routine should immediately write the transmit shifter register with 0x00 (if generating STOP) or 0xFF (if generating repeated START). Software should then wait for the next rising edge on SCL and then disable both timers. The transmit shifter should then be disabled after waiting the setup delay for a repeated START or STOP condition.

NOTE

Due to synchronization delays, the data valid time for the transmit output is 2 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

The I2C master data valid is delayed 2 cycles because the clock output is passed through a synchronizer before clocking the transmit/receive shifter (to guarantee some SDA hold time). Since the SCL output is synchronous with FlexIO clock, the synchronization delay is 1 cycle and then 1 cycle to generate the output.

Table 403. I2C Master Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Start bit enabled (logic 0) and stop bit enabled (logic 1).

Table continues on the next page...

Table 403. I2C Master Configuration (continued)

Register	Value	Comments
SHIFTCTLn	0x0101_0082	Configure transmit using Timer 1 on rising edge of clock with inverted output enable (open drain output) on Pin 0.
SHIFTCFG(n+1)	0x0000_0020	Start bit disabled and stop bit enabled (logic 0) for ACK/NACK detection.
SHIFTCTL(n+1)	0x0180_0001	Configure receive using Timer 1 on falling edge of clock with input data on Pin 0.
TIMCMPn	0x0000_2501	Configure 2 word transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of words x 18) + 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0102_2222	Configure start bit, stop bit, enable on trigger high, disable on compare, reset if output equals pin. Initial clock state is logic 0 and is not affected by reset.
TIMCTLn	0x01C1_0101	Configure dual 8-bit counter using Pin 1 output enable (SCL open drain), with Shifter 0 flag as the inverted trigger.
TIMCMP(n+1)	0x0000_000F	Configure 8-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG(n+1)	0x0020_1112	Enable when Timer 0 is enabled, disable when Timer 0 is disabled, enable start bit and stop bit at end of each word, decrement on pin input.
TIMCTL(n+1)	0x01C0_0183	Configure 16-bit counter using inverted Pin 1 input (SCL).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBBS[7:0], use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS[7:0], use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

69.4.6 I2S Master

I2S master mode can be supported using two Timers, two Shifters and four Pins. One timer is used to generate the bit clock and control the shifters and one timer is used to generate the frame sync. FlexIO waits for the first write to the transmit data buffer before enabling bit clock and frame sync generation. Data transfers can be supported using the DMA controller and the shifter error flag sets on transmit underrun or receive overflow.

The bit clock frequency is an even integer divide of the FlexIO clock frequency, and the initial frame sync assertion occurs at the same time as the first bit clock edge. The timer uses the start bit to ensure the frame sync is generated one clock cycle before the first output data.

NOTE

Due to synchronization delays, the setup time for the receiver input is 1.5 FlexIO clock cycles, so the maximum baud rate is divide by 4 of the FlexIO clock frequency.

Table 404. I2S Master Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0001	Load transmit data on first shift and stop bit disabled.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on rising edge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0080_0101	Configure receive using Timer 0 on falling edge of clock with input data on Pin 1.
TIMCMPn	0x0000_3F01	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0000_0202	Configure start bit, enable on trigger high and never disable. Initial clock state is logic 1.
TIMCTLn	0x01C3_0281	Configure dual 8-bit counter using inverted Pin 2 output (bit clock), with Shifter 0 flag as the inverted trigger. Clear PINPOL to invert the polarity of the output shift clock.
TIMCMP(n+1)	0x0000_007F	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:0] = (number of bits x baud rate divider) - 1.
TIMCFG(n+1)	0x0000_0100	Enable when Timer 0 is enabled and never disable.
TIMCTL(n+1)	0x0003_0383	Configure 16-bit counter using inverted Pin 3 output (as frame sync). Clear PINPOL to invert the polarity of the output frame sync
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.

Table continues on the next page...

Table 404. I2S Master Configuration (continued)

Register	Value	Comments
		Can support LSB first transfer by writing to SHIFTBUF register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support LSB first transfer by reading from SHIFTBUF register instead.

69.4.7 I2S Slave

I2S slave mode can be supported using three Timers, two Shifters and four Pins (for single transmit and single receive, other combinations of transmit and receive are possible).

The transmit data must be written to the transmit buffer register before the external frame sync asserts, otherwise the shifter error flag is set.

NOTE

Due to synchronization delays, the output valid time for the serial output data is 2.5 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

The output valid time of I2S slave is max 2.5 cycles because there is a maximum 1.5 cycle delay on the clock synchronization plus 1 cycle to output the data

Timer 2 detects the falling edge of frame sync (start of new frame) and asserts output until rising edge of bit clock (when frame sync is normally sampled). Timer 0 detects rising edge of bit clock with Timer 2 output asserted and asserts output for length of frame. Timer 1 detects falling edge of bit clock with Timer 0 output asserted and controls shift registers for 32-bit transfers.

Table 405. I2S Slave Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0000	Start and stop bit disabled.
SHIFTCTLn	0x0103_0002	Configure transmit using Timer 1 on rising edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0180_0101	Configure receive using Timer 1 on falling edge of shift clock with input data on Pin 1.
TIMCMPn	0x0000_007F	Configure two 32-bit transfers per frame. Set TIMCMP[15:0] = (number of bits x 4) - 1.
TIMCFGn	0x0020_2500	Configure enable on pin rising edge (inverted bit clock) with trigger high (Timer 2) and disable on compare, initial clock state is logic 1 and decrement on pin input (bit clock).

Table continues on the next page...

Table 405. I2S Slave Configuration (continued)

Register	Value	Comments
TIMCTLn	0x0B40_0203	Configure 16-bit counter using Pin 2 input (bit clock), with Timer 2 output as the trigger.
TIMCMP(n+1)	0x0000_003F	Configure 32-bit transfers. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG(n+1)	0x0020_2500	Configure enable on pin (bit clock) rising edge with trigger (Timer 0) high and disable on compare, initial clock state is logic 1 and decrement on pin input (bit clock).
TIMCTL(n+1)	0x0340_0283	Configure 16-bit counter using inverted Pin 2 input (bit clock), with Timer 0 output as the trigger.
TIMCMP(n+2)	0x0000_0000	Compare on zero (first edge).
TIMCFG(n+2)	0x0020_6400	Configure enable on inverted pin (frame sync) rising edge and disable on trigger falling edge (bit clock), initial clock state is logic 1 and decrement on inverted pin input (frame sync).
TIMCTL(n+2)	0x04C0_0383	Configure 16-bit counter using inverted Pin 3 input (frame sync), with Pin 2 inverted input (bit clock) as the trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support LSB first transfer by writing to SHIFTBUF register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support LSB first transfer by reading from SHIFTBUF register instead.

69.4.8 SENT Receiver

SENT receiver can be supported by using one Timer and one Pin. The Timer is configured as Input-Capture mode, and captures the conter value at falling edge of the Pin input. Once the counter value is captured, the counter is automatically restarted from counter value 0. Therefore, the captured value is always indicating the period between previous falling edge and current falling edge. CPU interrupt or DMA trigger can be configured at each capture of the counter, and the entire SENT frame decoding with the latest tick width adustment is performed by CPU software.

Table 406. SENT Receiver Configuration

Register	Value	Comments
TIMCMPn	-	Counter value at the falling edge of the Pin is stored in this register.
TIMCFGn	0x0000_6000	Timer always enabled, Timer disabled on Trigger falling edge, Decrement counter on FlexIO clock.
TIMCTLn	0xnn40_0007	Single 16-bit input capture mode, Timer pin input and output are selected by PINSEL, Timer pin output disabled, Internal trigger selected, Select Pin nn as Trigger.

69.4.9 Camera Interface

Camera Interface can be supported using one Timer, one or more Shifters and multiple Pins. Multiple transfers can be supported using DMA controller.

The example below describes FlexIO configuration for interfacing to an 8-bit CMOS sensor with PCLK, VSYNC, HREF and D[7:0] outputs. The example uses a 128-bit buffer to capture 16-pixels of image data before interrupt or DMA transfer, however a bigger or smaller buffer may be used depending on system DMA performance and FlexIO resource usage by other applications. Note that additional timers may be used to track number of pixels per row and number of rows per frame or HREF/VSYNC may be assigned as GPIO interrupts for software tracking.

Table 407. Camera Interface Configuration for 8-bit CMOS sensor

Register	Value	Comments
SHIFTCFGn...n+2[1]	0x0007_0100	Configure 8-bit parallel shift in from adjacent shifter.
SHIFTCFGn+3	0x0007_0000	Configure 8-bit parallel shift in from pins FXIO_D[7:0] (D[7:0]).
SHIFTCTLn...n+3	0x0080_0001	Configure receive using Timer 0 on negedge of clock.
TIMCMPn	0x0000_001F	Configure 16-pixel (8 bits/pixel x 16 pixels = 128-bits) transfer. Set TIMCMP[15:0] = (number of pixels x 2) - 1.
TIMCFGn	0x0120_6600	Configure enable on trigger (HREF) rising edge and disable on trigger falling edge, initial Shift clock state is logic 0 and decrement on PCLK input.
TIMCTLn	0x12C0_0803	Configure 16-bit counter using FXIO_D[8] input (PCLK), with FXIO_D[9] input (HREF) as the inverted trigger.
SHIFTBUFn...n+3	Data to receive	Received data can be read from SHIFTBUFn...n+3, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

1. n=0 or 4

69.4.10 Motorola 68K/Intel 8080 Bus Interface

The Motorola 68K and Intel 8080 bus are two parallel interfaces commonly used by Smart/Asynchronous LCD controllers. In conjunction with GPIO, FlexIO is able to drive these interfaces using one Timer and one Shifter, although additional Shifters could be used to support large transfers via the DMA controller.

The configuration below provides an example of how to drive a 16-bit 68K or 8080 bus. For a 8080 bus, two GPIO are used to drive the nCS and RS pins. For a 68K bus, an additional GPIO is required to drive the RDWR pin.

Table 408. Motorola 68K/Intel 8080 Write Configuration

Register	Value	Comments
SHIFTCFG0...7	0x000F_0100	Configure 16-bit parallel shift in from adjacent shifter.
SHIFTCTL0	0x0003_0002	Configure transmit using Timer 0 on posedge of clock with data output to FXIO_D[15:0].
SHIFTCTL1...7	0x0000_0002	Configure transmit using Timer 0 on posedge of clock.
TIMCMP0	0x0000_0101 (1-beat) 0x0000_1F01 (16-beats)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of beats x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0x0000_2200	Configure enable on trigger high and disable on compare. Timer output high on enable.
TIMCTL0	0x01C3_1001 (Motorola 68K, 1-beat) 0x1DC3_1001 (Motorola 68K, 16-beats) 0x01C3_1081 (Intel 8080, 1-beat) 0x1DC3_1081 (Intel 8080, 16-beats)	Configure dual 8-bit counter using FXIO_D[16] output (EN pin for 68K, nWR pin for 8080), with Shifter 0 (1-beat) or Shifter 7 (16-beats) flag as the inverted trigger.
SHIFTBUF0...7	Data to transmit	Transmit data can be written to SHIFTBUF0 (1-beat) or SHIFTBUF0...7 (16-beats) to initiate a transfer, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.

Table 409. Motorola 68K/Intel 8080 Read Configuration

Register	Value	Comments
SHIFTCFG0...6	0x000F_0100	Configure 16-bit parallel shift in from adjacent shifter.
SHIFTCFG7	0x000F_0000	Configure 16-bit parallel shift in from pin.

Table continues on the next page...

Table 409. Motorola 68K/Intel 8080 Read Configuration (continued)

Register	Value	Comments
SHIFTCTL0...7	0x0080_0001	Configure receive using Timer 0 on negedge of clock with data input from FXIO_D[15:0].
TIMCMP0	0x0000_0101 (1-beat) 0x0000_1F01 (16-beats)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of beats x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0x0000_2220	Configure stop_bit, enable on trigger high and disable on compare. Timer output high on enable.
TIMCTL0	0x1DC3_1001 (Motorola 68K, 1 beat) 0x01C3_1001 (Motorola 68K, 16 beats) 0x1DC3_1181 (Intel 8080, 1 beat) 0x01C3_1181 (Intel 8080, 16 beats)	Configure dual 8-bit counter using either FXIO_D[16] output (EN pin for 68K) or FXIO_D[17] output (nRD pin for 8080), with Shifter 7 flag (1-beat) or Shifter 0 flag (16-beats) as the inverted trigger.
SHIFTBUF0...7	Data received	Received data can be read from SHIFTBUF0 (1-beat) or SHIFTBUF0...7 (16-beats), use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

In general, any operation to a 68K/8080 bus slave begins with a register write cycle followed by one or more data read or write cycles. To accomplish this, the following program flow should be used:

1. Configure FlexIO with 1-beat write configuration
2. Configure GPIO to assert nCS, RS pins (and deassert RDWR pin for 68K)
3. Write register index data to SHIFTBUF0[15:0]
4. Configure GPIO to deassert RS pin (and assert RDWR pin for 68K data read)
5. Configure FlexIO with desired read or write configuration (e.g. 1 or 16-beats)
6. Use the Shifter Status Flag to trigger interrupt or DMA driven data transfers to/from SHIFTBUF registers
7. Configure GPIO to deassert nCS pin

69.4.11 Low Power State Machine

The configuration below details a hypothetical state machine example to illustrate the flexibility allowed when using Shifter state mode.

In this example, FlexIO waits for the FXIO_D[2] pin to assert and then drives a complementary square wave output at a frequency of FLEXIO_CLK/131072 on the FXIO_D[1:0] pins while the comparator output is asserted (This assumes comparator is connected to external trigger 15. See the chip-specific FlexIO information for actual FlexIO trigger mappings). The state diagram below shows the states and transitions implemented by this example.

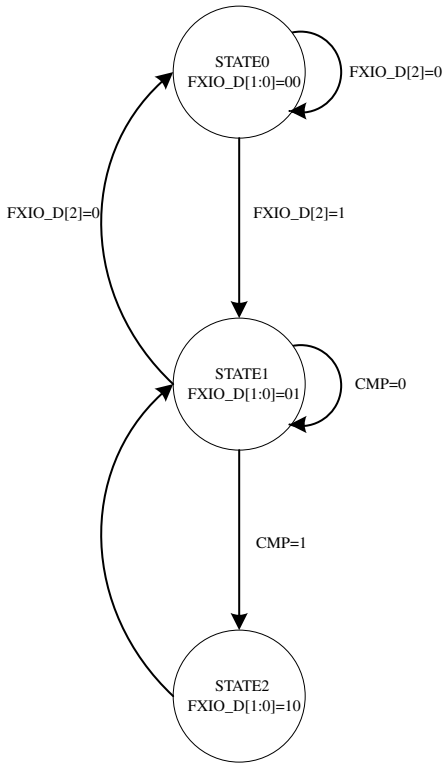


Figure 371. State Diagram

Table 410. State Machine Configuration

Register	Value	Comments
SHIFTCFG0...2	0x0000_0003	Enable FXIO_D[1:0] as outputs.
SHIFTCTL0	0x0080_0206	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer0 output low to trigger next state.
SHIFTBUF0	0x0020_8208	State0: Drive FXIO_D[1:0]=00, transition to State0 if FXIO_D[2]=0, State1 if FXIO_D[2]=1.
SHIFTCTL1	0x0000_0206	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer0 output high to trigger next state.
SHIFTBUF1	0x0140_8408	State1: Drive FXIO_D[1:0]=01, transition to State0 if FXIO_D[2]=0, State1 if CMP=0, State2 if CMP=1 (FXIO_D[3]=1)

Table continues on the next page...

Table 410. State Machine Configuration (continued)

Register	Value	Comments
SHIFTCTL2	0x0080_0206	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer0 output low to trigger next state.
SHIFTBUF2	0x0224_9249	State2: Drive FXIO_D[1:0]=10, transition to State1 when Timer0 output low
TIMCMP0	0x0000_FFFF	Configure baud rate of divide by 131072 of the FlexIO clock. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0x0000_0000	Configure timer always enabled.
TIMCTL0	0x0000_0003	Configure single 16-bit counter.
TIMCFG1	0x0010_7600	Configure timer enabled on trigger rising edge, disabled on trigger falling edge, decrement on trigger edge.
TIMCTL1	0x0F03_0303	Configure timer output enabled on FXIO_D[3], external trigger 15 (comparator output) selected.

69.4.12 Keypad Interface

The Keypad Interface can support a 3x4 keypad matrix using 3 timers and 3 shifters, although a larger matrix could be supported using additional shifters. The configuration is designed for 4 columns which are configured as active low open drain pins, and 3 rows which are configured as input pins with pullup resistors enabled.

One shifter is configured in logic mode and is configured to assert its output when any of the row inputs are low, indicating a key is pressed. A timer is used to filter the shifter output to ensure the key is pressed for a minimum amount of time before performing the column scan.

Another shifter is configured for parallel transmit, this shifter is used to scan each column when a key press is detected. Whenever not scanning, the shifter output is configured to assert all active low open drain column outputs to detect any keypress. A dedicated timer is used to control the transmit shifter.

The last shifter is configured for parallel receive, this shifter is used to capture the result of the column scanning for software to decode which key or keys was pressed. This configuration captures the state of both row and colum pins for each scan, although the row state could also be deduced by the shift order. A dedicated timer is used to control the receive shifter, it shifts at half the frequency of the transmit shifter.

Once the result of the key scan is available in the receive shifter register, FlexIO continues to monitor the row inputs and can trigger multiple scans from the one key press. To support deboucing, software can decide how many consecutive scans should be considered a single key press.

Since the pins used in logic mode are fixed per shifter, and the shifters that support parallel shifts are limited, this configuration is restricted in what pins and shifters it can use. Increasing the matrix beyond 4 row inputs requires multiple shifters in logic mode and increasing beyond 4 column outputs requires concatenating transmit shifters to create a larger shift register.

Table 411. Keypad Interface Configuration

Register	Value	Comments
SHIFTCFG0	0x0003_0032	Start bit enabled (logic 0) and stop bit enabled (logic 1), configired for 4-bit shift.
SHIFTCTL0	0x0101_0402	Configure transmit using Timer 1 on rising edge of clock generating open drain output on Pins [7:4] (column outputs).
SHIFTBUF0	0x0804_0201	Static data containing the column scan pattern, each column is scanned one-hot with deadtime inbetween.
SHIFTCFG1	0x0000_0020	In logic mode, mask input 3.
SHIFTCTL1	0x0000_0007	Configure logic mode.
SHIFTBUF1	0x07ff_07ff	Static data configuring logic mode LUT. Output asserts if pins [3:1] are logic 0.
SHIFTCFG3	0x0007_0000	Configured for 8-bit shift.
SHIFTCTL3	0x0280_0001	Configure receive using Timer 2 on falling edge of clock with input data on Pins [7:0] (both rows and columns, pin 0 is don't care).
TIMCMP0	0x0000_00ff	Configures pre-scanning glitch filter to 256 FlexIO clock cycles. For different filter cycles, set TIMCMP[15:0] = (filter cycles) - 1.
TIMCFG0	0x0103_6600	Configure enable on trigger rising edge and disable on trigger falling edge. Initial clock state is logic 0 and is not affected by reset.
TIMCTL0	0x0540_0003	Configure 16-bit counter using Shifter 1 flag (logic state) as trigger.
TIMCMP1	0x0000_0f3f	Configure 8 shifts (twice for each column) at column scan rate of divide by 128. For different scan frequency, set TIMCMP[7:0] = (scan divider / 2) - 1.
TIMCFG1	0x0020_2622	Enable on trigger rising edge, disable on compare, start and stop bits are enabled.
TIMCTL1	0x0340_0001	Configure dual 8-bit counter using Timer 0 output as the trigger.
TIMCMP2	0x0000_0801	Configure 4 shifts at half the frequency of Timer 1 trigger.
TIMCFG2	0x0110_2100	Enable on Timer 1 enable, disable on compare, decrement on trigger input

Table continues on the next page...

Table 411. Keypad Interface Configuration (continued)

Register	Value	Comments
		with output initially negated and not affected by reset
TIMCTL2	0x0740_0001	Configure dual 8-bit counter using Timer 1 output as the trigger.
SHIFTBUF3	Keypad Scan Result	Keypad scan result can be read from SHIFTBUF3, each byte is the result of one scan with both row and column pin state from the scan (note that pin 0 is not used). Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.

69.5 FlexIO Signal Descriptions

Table 412.

Signal	Description	I/O
FXIO_Dn (n=0...31)	Bidirectional FlexIO Shifter and Timer pin inputs/outputs	I/O

69.6 Memory Map and Registers

69.6.1 FLEXIO register descriptions

NOTE

An invalid register access results in a bus error. This includes reading a write-only register, writing a read-only register or accessing an invalid address.

69.6.1.1 FLEXIO memory map

FlexIO base address: 4032_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	R	0201_0003h
4h	Parameter Register (PARAM)	32	R	0420_0808h
8h	FlexIO Control Register (CTRL)	32	RW	0000_0000h
Ch	Pin State Register (PIN)	32	R	0000_0000h
10h	Shifter Status Register (SHIFTSTAT)	32	RW	0000_0000h
14h	Shifter Error Register (SHIFTERR)	32	RW	0000_0000h
18h	Timer Status Register (TIMSTAT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20h	Shifter Status Interrupt Enable (SHIFTSIEN)	32	RW	0000_0000h
24h	Shifter Error Interrupt Enable (SHIFTEIEN)	32	RW	0000_0000h
28h	Timer Interrupt Enable Register (TIMIEN)	32	RW	0000_0000h
30h	Shifter Status DMA Enable (SHIFTSDEN)	32	RW	0000_0000h
38h	Timer Status DMA Enable (TIMERSDEN)	32	RW	0000_0000h
40h	Shifter State Register (SHIFTSTATE)	32	RW	0000_0000h
48h	Trigger Status Register (TRGSTAT)	32	RW	0000_0000h
4Ch	External Trigger Interrupt Enable Register (TRIGIEN)	32	RW	0000_0000h
50h	Pin Status Register (PINSTAT)	32	RW	0000_0000h
54h	Pin Interrupt Enable Register (PINIEN)	32	RW	0000_0000h
58h	Pin Rising Edge Enable Register (PINREN)	32	RW	0000_0000h
5Ch	Pin Falling Edge Enable Register (PINFEN)	32	RW	0000_0000h
60h	Pin Output Data Register (PINOUTD)	32	RW	0000_0000h
64h	Pin Output Enable Register (PINOUTE)	32	RW	0000_0000h
68h	Pin Output Disable Register (PINOUTDIS)	32	W	0000_0000h
6Ch	Pin Output Clear Register (PINOUTCLR)	32	W	0000_0000h
70h	Pin Output Set Register (PINOUTSET)	32	W	0000_0000h
74h	Pin Output Toggle Register (PINOUTTOG)	32	W	0000_0000h
80h - 9Ch	Shifter Control N Register (SHIFTCTL0 - SHIFTCTL7)	32	RW	0000_0000h
100h - 11Ch	Shifter Configuration N Register (SHIFTCFG0 - SHIFTCFG7)	32	RW	0000_0000h
200h - 21Ch	Shifter Buffer N Register (SHIFTBUF0 - SHIFTBUF7)	32	RW	0000_0000h
280h - 29Ch	Shifter Buffer N Bit Swapped Register (SHIFTBUFBIS0 - SHIFTBUFBIS7)	32	RW	0000_0000h
300h - 31Ch	Shifter Buffer N Byte Swapped Register (SHIFTBUFBYS0 - SHIFTBUFBYS7)	32	RW	0000_0000h
380h - 39Ch	Shifter Buffer N Bit Byte Swapped Register (SHIFTBUFBBS0 - SHIFTBUFBBS7)	32	RW	0000_0000h
400h - 41Ch	Timer Control N Register (TIMCTL0 - TIMCTL7)	32	RW	0000_0000h
480h - 49Ch	Timer Configuration N Register (TIMCFG0 - TIMCFG7)	32	RW	0000_0000h
500h - 51Ch	Timer Compare N Register (TIMCMP0 - TIMCMP7)	32	RW	0000_0000h
680h - 69Ch	Shifter Buffer N Nibble Byte Swapped Register (SHIFTBUFNBS0 - SHIFTBUFNBS7)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
700h - 71Ch	Shifter Buffer N Half Word Swapped Register (SHIFTBUFHWS0 - SHIFTBUFHWS7)	32	RW	0000_0000h
780h - 79Ch	Shifter Buffer N Nibble Swapped Register (SHIFTBUFNIS0 - SHIFTBUFNIS7)	32	RW	0000_0000h
800h - 81Ch	Shifter Buffer N Odd Even Swapped Register (SHIFTBUFOES0 - SHIFTBUFOES7)	32	RW	0000_0000h
880h - 89Ch	Shifter Buffer N Even Odd Swapped Register (SHIFTBUFEOS0 - SHIFTBUFEOS7)	32	RW	0000_0000h
900h - 91Ch	Shifter Buffer N Halfword Byte Swapped Register (SHIFTBUFHBS0 - SHIFTBUFHBS7)	32	RW	0000_0000h

69.6.1.2 Version ID Register (VERID)

Offset

Register	Offset
VERID	0h

Function

Contains the version of the FlexIO.

Diagram

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

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

Fields

Field	Function
31-24	Major Version Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
MAJOR	This read only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000_0000_0000_0000b - Standard features implemented. 0000_0000_0000_0001b - Supports state, logic and parallel modes. 0000_0000_0000_0010b - Supports pin control registers. 0000_0000_0000_0011b - Supports state, logic and parallel modes; plus pin control registers.

69.6.1.3 Parameter Register (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Contains the number of shifters, timers, pins, and triggers.

Diagram

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

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

Fields

Field	Function
31-24	Trigger Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRIGGER	Number of external triggers implemented.
23-16 PIN	Pin Number Number of Pins implemented.
15-8 TIMER	Timer Number Number of Timers implemented.
7-0 SHIFTER	Shifter Number Number of Shifters implemented.

69.6.1.4 FlexIO Control Register (CTRL)

Offset

Register	Offset
CTRL	8h

Function
Control register.

Diagram

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

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

Fields

Field	Function
31 —	Any write value to this bit must be zero.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 DBGE	Debug Enable Enables FlexIO operation in Debug mode. 0b - FlexIO is disabled in debug modes. 1b - FlexIO is enabled in debug modes
29-3 —	Reserved
2 FASTACC	Fast Access Enables fast register accesses to FlexIO registers, but requires the FlexIO functional clock to be at least equal to the frequency of the bus clock. 0b - Configures for normal register accesses to FlexIO 1b - Configures for fast register accesses to FlexIO
1 SWRST	Software Reset The FlexIO Control Register is not affected by the software reset, all other logic in the FlexIO is affected by the software reset and all other register accesses are ignored until this bit is cleared. This register bit remains set until cleared by software, and the reset has cleared in the FlexIO clock domain. 0b - Software reset is disabled 1b - Software reset is enabled, all FlexIO registers except the Control Register are reset.
0 FLEXEN	FlexIO Enable Determines if the FlexIO is enabled or disabled. 0b - FlexIO module is disabled. 1b - FlexIO module is enabled.

69.6.1.5 Pin State Register (PIN)

Offset

Register	Offset
PIN	Ch

Function
Reports status of the Pin Data Input.

Diagram

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

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

Fields

Field	Function
31-0 PDI	Pin Data Input Returns the input data on each of the FlexIO pins.

69.6.1.6 Shifter Status Register (SHIFTSTAT)

Offset

Register	Offset
SHIFTSTAT	10h

Function

Shifter status flags.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								SSF							
W									W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 SSF	Shifter Status Flag The shifter status flag is updated when one of the following events occurs: For SMOD=Receive, the status flag is set when SHIFTBUF has been loaded with data from Shifter (SHIFTBUF is full), and the status flag is cleared when SHIFTBUF register is read. For SMOD=Transmit, the status flag is set when SHIFTBUF data has been transferred to the Shifter (SHIFTBUF is empty) or when initially configured for SMOD=Transmit, and the status flag is cleared when the SHIFTBUF register is written. For SMOD=Match Store, the status flag is set when a match has occured between SHIFTBUF and Shifter, and the status flag is cleared when the SHIFTBUF register is read. For SMOD=Match Continuous, returns the current match result between the SHIFTBUF and Shifter. The status flag cannot be cleared by reading SHIFTBUF. For SMOD=State, the status flag for a shifter sets when it is selected by the current state pointer. For SMOD=Logic, returns the current value of the programmable logic block output. The status flag can also be cleared by writing a logic one to the flag for all modes except Match Continuous/ State /Logic. 0b - Status flag is clear. 1b - Status flag is set.

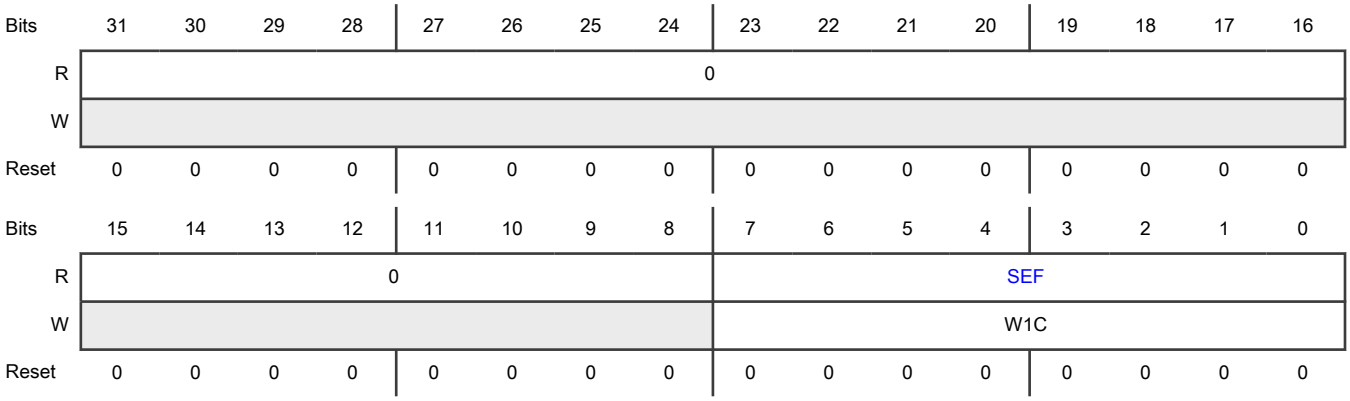
69.6.1.7 Shifter Error Register (SHIFTERR)

Offset

Register	Offset
SHIFTERR	14h

Function
This register reports shifter error flags. Write one to clear.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SEF	Shifter Error Flags The shifter error flag is set when one of the following events occurs: For SMOD=Receive, indicates Shifter was ready to store new data into SHIFTBUF before the previous data was read from SHIFTBUF (SHIFTBUF Overrun), or indicates that the received start or stop bit does not match the expected value. For SMOD=Transmit, indicates Shifter was ready to load new data from SHIFTBUF before new data had been written into SHIFTBUF (SHIFTBUF Underrun). For SMOD=Match Store, indicates a match event occured before the previous match data was read from SHIFTBUF (SHIFTBUF Overrun). For SMOD=Match Continuous, the error flag is set when a match has occured between SHIFTBUF and Shifter. For SMOD=Logic, the error flag is set when the output of the programmable logic block has asserted. Can be cleared by writing logic one to the flag. For SMOD=Match Continuous, can also be cleared when the SHIFTBUF register is read. 0b - Shifter Error Flag is clear. 1b - Shifter Error Flag is set.

69.6.1.8 Timer Status Register (TIMSTAT)

Offset

Register	Offset
TIMSTAT	18h

Function

This register reports timer status flags. Write one to clear.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TSF							
W									W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 TSF	Timer Status Flags The timer status flag sets depending on the timer mode, and can be cleared by writing logic one to the flag. In 8-bit baud counter mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 8-bit high PWM mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 16-bit counter mode, the timer status flag is set when the 16-bit counter equals zero and decrements. In 16-bit counter disable mode, the timer status flag is set when a timer disable event is detected. In 8-bit word counter mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 8-bit low PWM mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 16-bit input capture mode, the timer status flag is set when a timer disable event is detected and the timer status flag is clear. In this mode, the timer compare register should only be read when the timer status flag is set. 0b - Timer Status Flag is clear. 1b - Timer Status Flag is set.

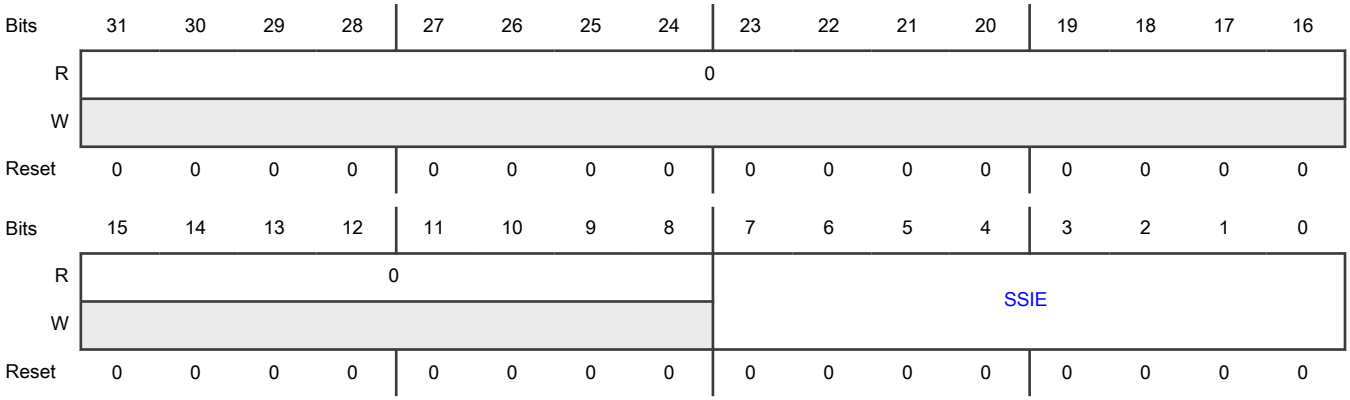
69.6.1.9 Shifter Status Interrupt Enable (SHIFTSIEN)

Offset

Register	Offset
SHIFTSIEN	20h

Function
Enables for Shifter Status Interrupts.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SSIE	Shifter Status Interrupt Enable Enables interrupt generation when corresponding SSF is set. 0b - Shifter Status Flag interrupt disabled. 1b - Shifter Status Flag interrupt enabled.

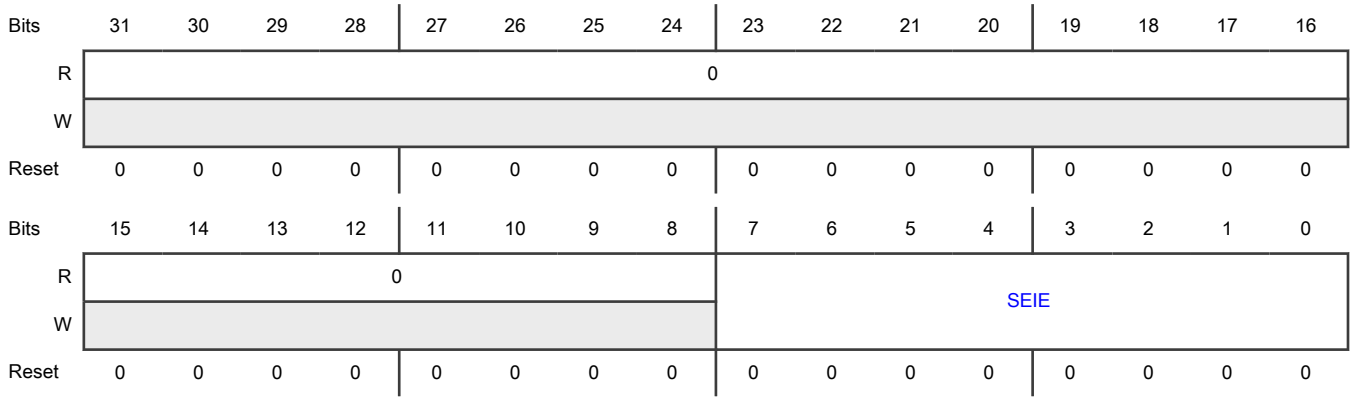
69.6.1.10 Shifter Error Interrupt Enable (SHIFTEIEN)

Offset

Register	Offset
SHIFTEIEN	24h

Function
Enables for Shifter Error Interrupts.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SEIE	Shifter Error Interrupt Enable Enables interrupt generation when corresponding SEF is set. 0b - Shifter Error Flag interrupt disabled. 1b - Shifter Error Flag interrupt enabled.

69.6.1.11 Timer Interrupt Enable Register (TIMIEN)

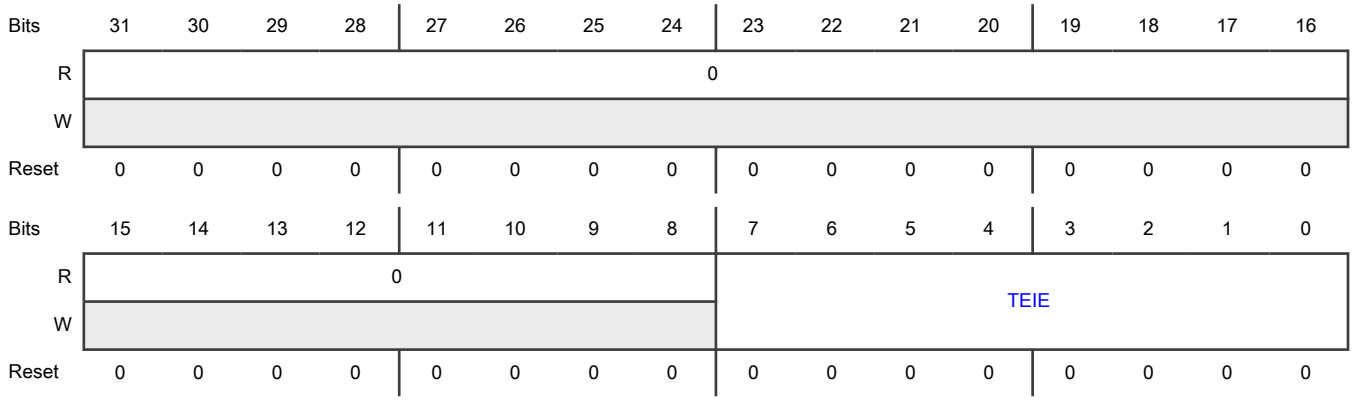
Offset

Register	Offset
TIMIEN	28h

Function

Enables for Timer Status Interrupts.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 TEIE	Timer Status Interrupt Enable Enables interrupt generation when corresponding TSF is set. 0b - Timer Status Flag interrupt is disabled. 1b - Timer Status Flag interrupt is enabled.

69.6.1.12 Shifter Status DMA Enable (SHIFTSDEN)

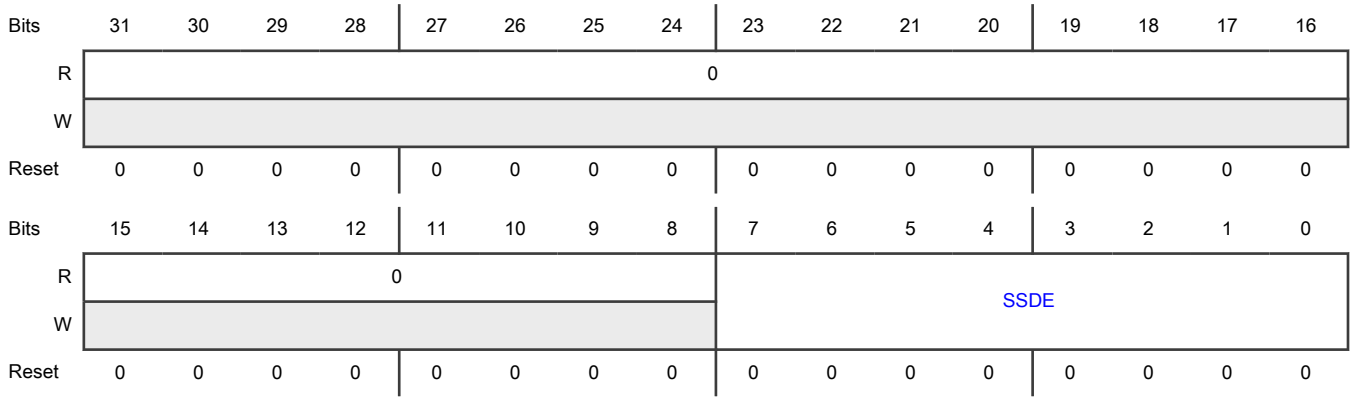
Offset

Register	Offset
SHIFTSDEN	30h

Function

Enables for Shifter DMA requests.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SSDE	Shifter Status DMA Enable Enables DMA request generation when corresponding SSF is set. 0b - Shifter Status Flag DMA request is disabled. 1b - Shifter Status Flag DMA request is enabled.

69.6.1.13 Timer Status DMA Enable (TIMERSDEN)

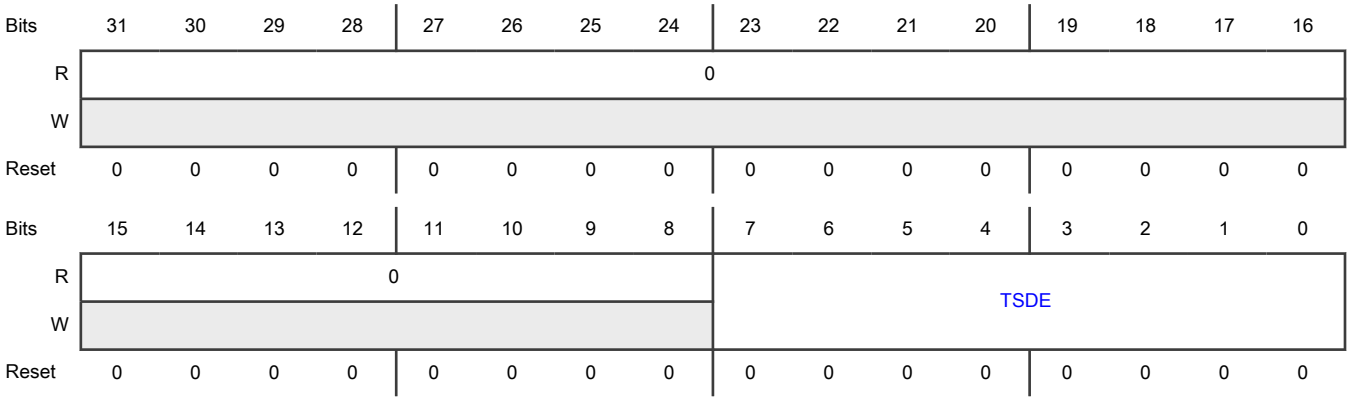
Offset

Register	Offset
TIMERSDEN	38h

Function

Enables for DMA requests when Timer Status Flag is set.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 TSDE	Timer Status DMA Enable Enables DMA request generation when corresponding TSF is set. When the Timer Status DMA request is enabled, reading or writing a Timer Compare Register clears the corresponding Timer Status Register. The DMA must therefore read or write the Timer Compare Register as part of the DMA transfer, otherwise the DMA request remains asserted. 0b - Timer Status Flag DMA request is disabled. 1b - Timer Status Flag DMA request is enabled.

69.6.1.14 Shifter State Register (SHIFTSTATE)

Offset

Register	Offset
SHIFTSTATE	40h

Function

Contains the pointer to keep track of the current shifter.

Diagram

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

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

Fields

Field	Function
31-3 —	Reserved
2-0 STATE	Current State Pointer The current state field maintains a pointer to keep track of the current Shifter (configured for State mode) enabled to drive outputs and compute the next state. Reading this register when the state pointer is updating can result in the incorrect state returned.

69.6.1.15 Trigger Status Register (TRGSTAT)

Offset

Register	Offset
TRGSTAT	48h

Function
External Trigger Status Flags. Write one to clear.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														ETSF	
W															W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 ETSF	External Trigger Status Flags The external trigger status flag is set when a rising edge is detected on the corresponding external trigger input. 0b - External Trigger Status Flag is clear 1b - External Trigger Status Flag is set

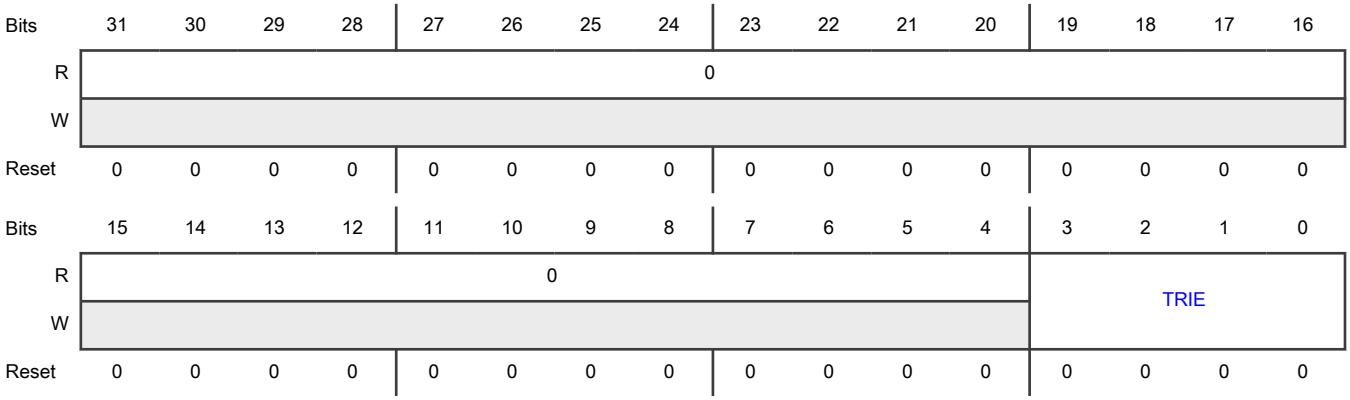
69.6.1.16 External Trigger Interrupt Enable Register (TRIGIEN)

Offset

Register	Offset
TRIGIEN	4Ch

Function
Enables for external trigger interrupts.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0	External Trigger Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRIE	Enables interrupt generation when corresponding ETSF is set. 0b - External Trigger Status Flag interrupt is disabled. 1b - External Trigger Status Flag interrupt is enabled.

69.6.1.17 Pin Status Register (PINSTAT)

Offset

Register	Offset
PINSTAT	50h

Function

Pin Status Flags. Write one to clear.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PSF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PSF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PSF	Pin Status Flags The pin status flag is set when a rising edge (if configured) or falling edge (if configured) is detected on the corresponding pin, as configured by pin. 0b - Pin Status Flag is clear 1b - Pin Status Flag is set

69.6.1.18 Pin Interrupt Enable Register (PINIEN)

Offset

Register	Offset
PINIEN	54h

Function
Enables for pin status interrupts.

Diagram

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

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

Fields

Field	Function
31-0 PSIE	Pin Status Interrupt Enable Enables interrupt generation when corresponding PSF is set. 0b - Pin Status Flag interrupt is disabled. 1b - Pin Status Flag interrupt is enabled.

69.6.1.19 Pin Rising Edge Enable Register (PINREN)

Offset

Register	Offset
PINREN	58h

Function
Enables for pin status flag on rising edge.

Diagram

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

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

Fields

Field	Function
31-0 PRE	Pin Rising Edge When set, the pin status flag is set whenever a rising edge is detected on the pin. 0b - Pin Status Flag does not set when a pin rising edge is detected. 1b - Pin Status Flag does set when a pin rising edge is detected.

69.6.1.20 Pin Falling Edge Enable Register (PINFEN)

Offset

Register	Offset
PINFEN	5Ch

Function
Enables for pin status flag on falling edge.

Diagram

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

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

Fields

Field	Function
31-0 PFE	Pin Falling Edge When set, the pin status flag is set whenever a falling edge is detected on the pin. 0b - Pin Status Flag does not set when a pin falling edge is detected. 1b - Pin Status Flag does set when a pin falling edge is detected.

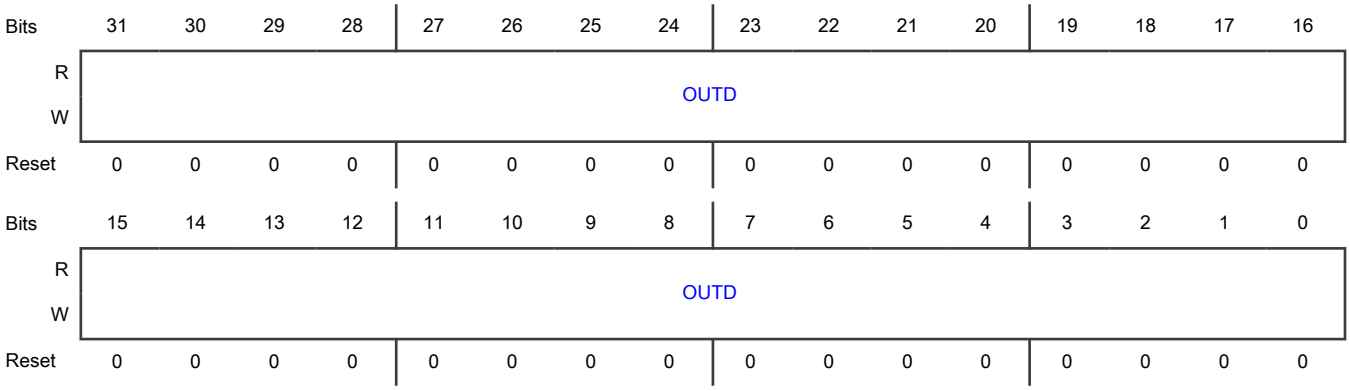
69.6.1.21 Pin Output Data Register (PINOUTD)

Offset

Register	Offset
PINOUTD	60h

Function
Data Out when direct pin output in enabled.

Diagram



Fields

Field	Function
31-0 OUTD	Output Data Configures the value driven on the corresponding pin when direct pin output is enabled. 0b - Logic zero is driven on the pin when direct pin output is enabled. 1b - Logic one is driven on the pin when direct pin output is enabled.

69.6.1.22 Pin Output Enable Register (PINOUTE)

Offset

Register	Offset
PINOUTE	64h

Function
Enables for pin output.

Diagram

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

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

Fields

Field	Function
31-0 OUTE	Output Enable Enables direct output on the corresponding pin. 0b - Pin is controlled by timer/shifter configuration. 1b - Pin is an output and driven with value of PINOUTD register.

69.6.1.23 Pin Output Disable Register (PINOUTDIS)

Offset

Register	Offset
PINOUTDIS	68h

Function
Disable for pin output.

Diagram

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

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

Fields

Field	Function
31-0 OUTDIS	Output Disable Configures correponding pins to disable direct output. 0b - No effect. 1b - Corresponding PINOUTD and PINOUTE bits are cleared.

69.6.1.24 Pin Output Clear Register (PINOUTCLR)

Offset

Register	Offset
PINOUTCLR	6Ch

Function
Clears pin output.

Diagram

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

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

Fields

Field	Function
31-0 OUTCLR	Output Clear Configures correponding pins to output zero. 0b - No effect. 1b - Corresponding PINOUTD is cleared and PINOUTE is set.

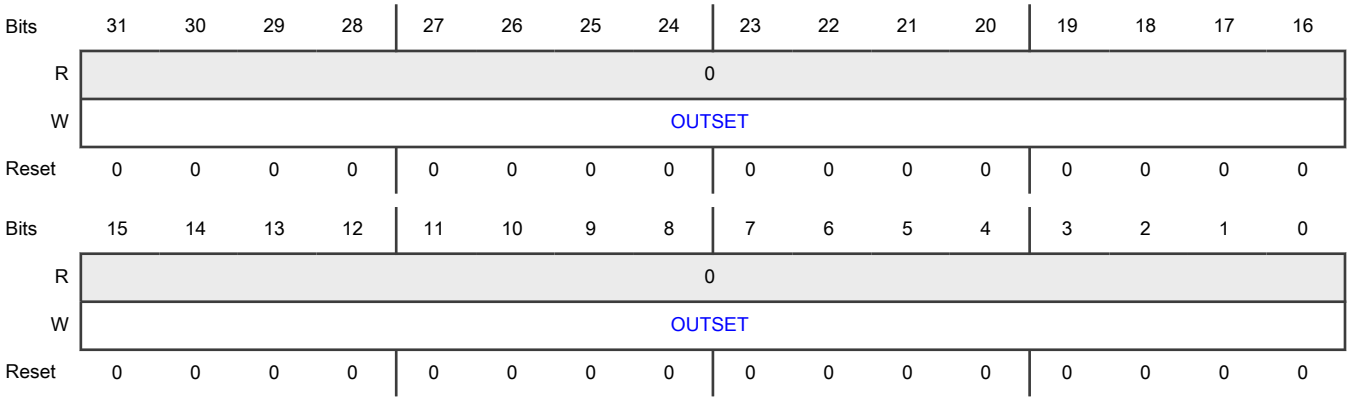
69.6.1.25 Pin Output Set Register (PINOUTSET)

Offset

Register	Offset
PINOUTSET	70h

Function
sets pin output.

Diagram



Fields

Field	Function
31-0 OUTSET	Output Set Configures correponding pins to output logic one. 0b - No effect. 1b - Corresponding PINOUTD is set and PINOUTE is set.

69.6.1.26 Pin Output Toggle Register (PINOUTTOG)

Offset

Register	Offset
PINOUTTOG	74h

Function
Toggles pin output.

Diagram

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

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

Fields

Field	Function
31-0 OUTTOG	Output Toggle Configures corresponding pins to toggle. 0b - No effect. 1b - Corresponding PINOUTD is inverted and PINOUTE is set.

69.6.1.27 Shifter Control N Register (SHIFTCTL0 - SHIFTCTL7)

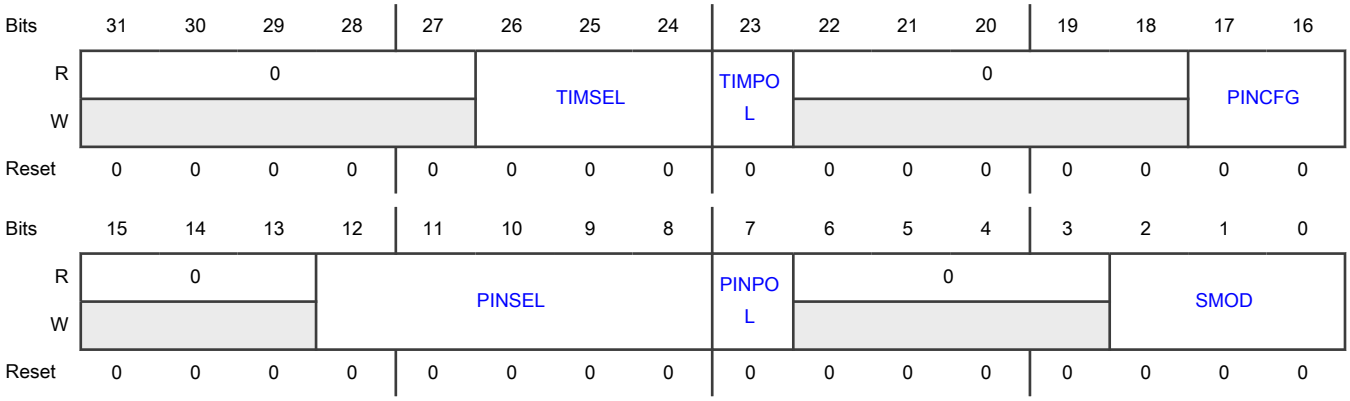
Offset

For a = 0 to 7:

Register	Offset
SHIFTCTLa	80h + (a × 4h)

Function
Control for Shift Register N

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 TIMSEL	Timer Select Selects which Timer is used for controlling the logic/shift register and generating the Shift clock. TIMSEL=i selects TIMERi.
23 TIMPOL	Timer Polarity Determines the shift occurs on the positive edge or negative edge of the shift clock. 0b - Shift on posedge of Shift clock 1b - Shift on negedge of Shift clock
22-18 —	Reserved
17-16 PINCFG	Shifter Pin Configuration For pins configured as an output (PINCFG=11), this field takes effect when the register is written. 00b - Shifter pin output disabled 01b - Shifter pin open drain or bidirectional output enable 10b - Shifter pin bidirectional output data 11b - Shifter pin output
15-13 —	Reserved
12-8 PINSEL	Shifter Pin Select Selects which pin is used by the Shifter input or output. PINSEL=i selects the FXIO_Di pin. For pins configured as an output (PINCFG=11), this field takes effect when the register is written.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 PINPOL	Shifter Pin Polarity For pins configured as an output (PINCFG=11), this field takes effect when the register is written. 0b - Pin is active high 1b - Pin is active low
6-3 —	Reserved
2-0 SMOD	Shifter Mode Configures the mode of the Shifter. 000b - Disabled. 001b - Receive mode. Captures the current Shifter content into the SHIFTBUF on expiration of the Timer. 010b - Transmit mode. Load SHIFTBUF contents into the Shifter on expiration of the Timer. 011b - Reserved. 100b - Match Store mode. Shifter data is compared to SHIFTBUF content on expiration of the Timer. 101b - Match Continuous mode. Shifter data is continuously compared to SHIFTBUF contents. 110b - State mode. SHIFTBUF contents are used for storing programmable state attributes. 111b - Logic mode. SHIFTBUF contents are used for implementing programmable logic look up table.

69.6.1.28 Shifter Configuration N Register (SHIFTCFG0 - SHIFTCFG7)

Offset

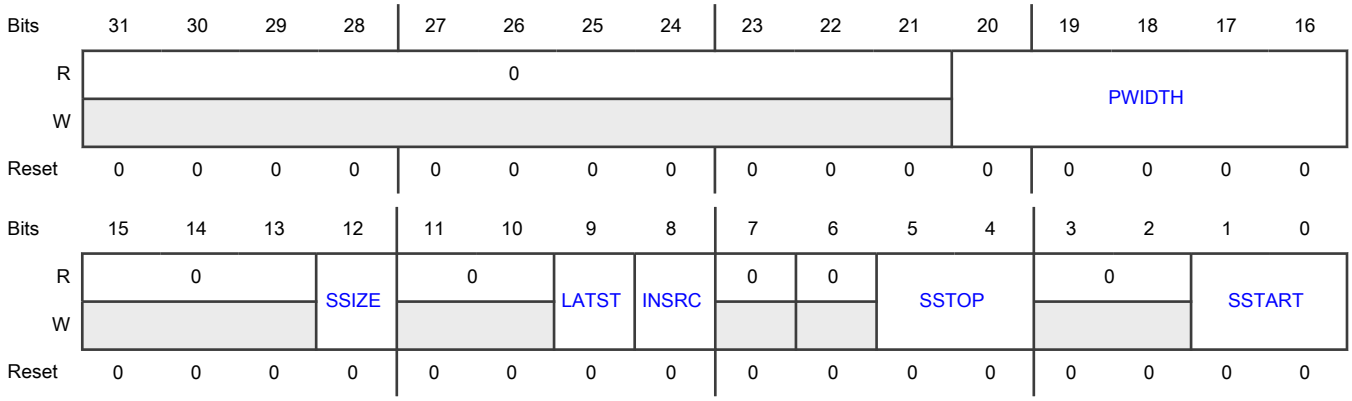
For a = 0 to 7:

Register	Offset
SHIFTCFGa	100h + (a × 4h)

Function

Contains configuration bit fields for Shifter Register N.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 PWIDTH	Parallel Width For all Shifters, this register field configures the number of bits to be shifted on each Shift clock as follows: 1-bit shift for PWIDTH=0 2-bit shift for PWIDTH=1 4-bit shift for PWIDTH=2...3 8-bit shift for PWIDTH=4...7 16-bit shift for PWIDTH=8...15 32-bit shift for PWIDTH=16...31 For Shifters which support parallel transmit (SHIFTER0, SHIFTER4, ...) or parallel receive (SHIFTER3, SHIFTER7, ...), this register field, together with SHIFTCTL[PINSEL], also selects the pins to be driven or sampled on each Shift clock as follows: • FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL] Shifters which do not support parallel transmit or parallel receive only support parallel shift when INSRC=1. For SMOD=State, this field is used to disable state outputs. See State Mode section for more detail.
15-13 —	Reserved
12 SSIZE	Shifter Size Configures the size of the Shift Registers. A 24-bit shift register shifts data into bits [23:0] only and does not update bits [31:24] during shift operations. When shift register is configured for 24-bit, configuring PWIDTH=8..15 performs a 12-bit shift and PWIDTH=16..31 performs a 24-bit shift.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Shift register is 32-bit. 1b - Shift register is 24-bit.
11-10 —	Reserved
9 LATST	Late Store Configures what happens when a receive or match shift register is configured to both shift and store on the same cycle. 0b - Shift register stores the pre-shift register state. 1b - Shift register stores the post-shift register state.
8 INSRC	Input Source Selects the input source for the shifter. Configuring INSRC=1 is not supported for the last shifter. 0b - Pin 1b - Shifter N+1 Output
7 —	Reserved
6 —	Reserved
5-4 SSTOP	Shifter Stop bit For SMOD=Transmit, this field allows automatic stop bit insertion if the selected timer has also enabled a stop bit. For SMOD=Receive or Match Store, this field allows automatic stop bit checking if the selected timer has also enabled a stop bit. For SMOD=State, this field is used to disable state outputs. See 'State Mode' section for more detail. For SMOD=Logic, this field is used to mask logic pin inputs. See 'Logic Mode' section for more detail. 00b - Stop bit disabled for transmitter/receiver/match store 01b - Reserved for transmitter/receiver/match store 10b - Transmitter outputs stop bit value 0 on store, receiver/match store sets error flag if stop bit is not 0 11b - Transmitter outputs stop bit value 1 on store, receiver/match store sets error flag if stop bit is not 1
3-2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1-0 SSTART	Shifter Start bit For SMOD=Transmit, this field allows automatic start bit insertion if the selected timer has also enabled a start bit. For SMOD=Receive or Match Store, this field allows automatic start bit checking if the selected timer has also enabled a start bit. For SMOD=State, this field is used to disable state outputs. See 'State Mode' section for more detail. For SMOD=Logic, this field is used to mask logic pin inputs. See 'Logic Mode' section for more detail. 00b - Start bit disabled for transmitter/receiver/match store, transmitter loads data on enable 01b - Start bit disabled for transmitter/receiver/match store, transmitter loads data on first shift 10b - Transmitter outputs start bit value 0 before loading data on first shift, receiver/match store sets error flag if start bit is not 0 11b - Transmitter outputs start bit value 1 before loading data on first shift, receiver/match store sets error flag if start bit is not 1

69.6.1.29 Shifter Buffer N Register (SHIFTBUF0 - SHIFTBUF7)

Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFa	200h + (a × 4h)

Function

Contains Shift buffer data.

Diagram

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

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

Fields

Field	Function
31-0 SHIFTBUF	Shift Buffer Shift buffer data is used for a variety of functions depending on the SMOD setting: For SMOD=Receive, Shifter data is transferred into SHIFTBUF at the expiration of Timer. The SHIFTBUF register should only be read when the corresponding shifter status flag is set, indicating new Shifter data is available. For SMOD=Transmit, SHIFTBUF data is transferred into the Shifter before the Timer begins. For SMOD=Match Store, SHIFTBUF[31:16] contains the data to be matched with the Shifter contents and SHIFTBUF[15:0] can be used to mask the match result (1=mask, 0=no mask). The Match is checked when the Timer expires and Shifter data [31:16] is written to SHIFTBUF[31:16] whenever a match event occurs. The SHIFTBUF register should only be read when the corresponding shifter status flag is set, indicating new Shifter data is available. For SMOD=Match Continuous, SHIFTBUF[31:16] contains the data to be matched with the Shifter contents and SHIFTBUF[15:0] can be used to mask the match result (1=mask, 0=no mask). For SMOD=Logic, SHIFTBUF[31:0] is used to implement a 5-input, 32-bit programmable logic look-up table. See Logic Mode section for more detail. For SMOD=State, SHIFTBUF[31:24] is used to drive the output value when this shifter is selected by the current state pointer and SHIFTBUF[23:0] is used to configure the value of the next state transition. See State Mode section for more detail.

69.6.1.30 Shifter Buffer N Bit Swapped Register (SHIFTBUFBIS0 - SHIFTBUFBIS7)

Offset

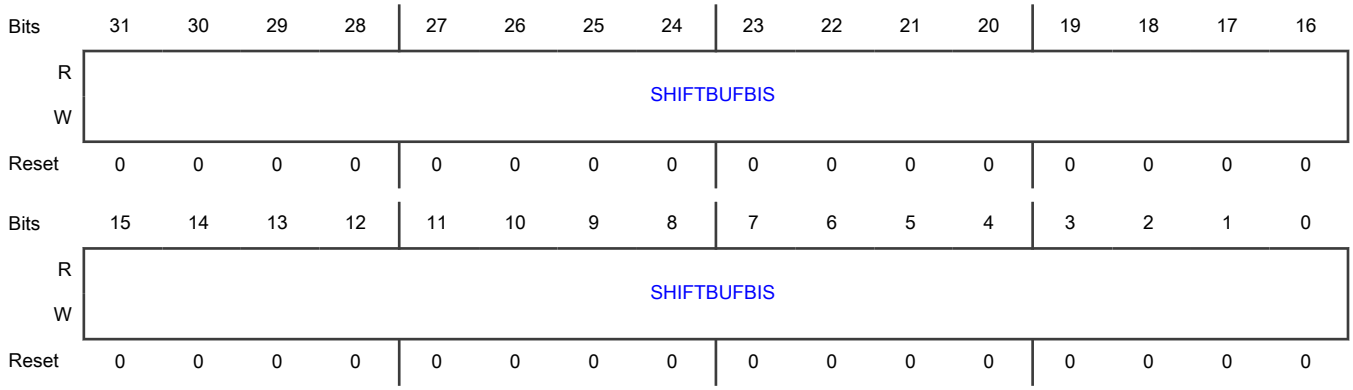
For a = 0 to 7:

Register	Offset
SHIFTBUFBISa	280h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register are bit swapped.

Diagram



Fields

Field	Function
31-0 SHIFTBUFBIS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are bit swapped. Reads return SHIFTBUF[0:31].

69.6.1.31 Shifter Buffer N Byte Swapped Register (SHIFTBUFBYS0 - SHIFTBUFBYS7)

Offset

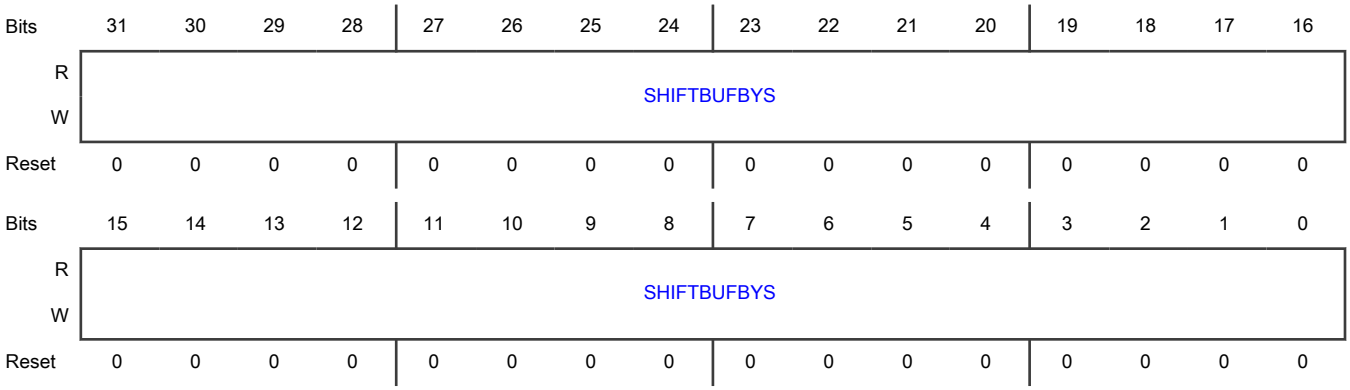
For a = 0 to 7:

Register	Offset
SHIFTBUFBYSa	300h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register are byte swapped.

Diagram



Fields

Field	Function
31-0 SHIFTBUFBYS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are byte swapped. Reads return { SHIFTBUF[7:0], SHIFTBUF[15:8], SHIFTBUF[23:16], SHIFTBUF[31:24] }.

69.6.1.32 Shifter Buffer N Bit Byte Swapped Register (SHIFTBUFBBS0 - SHIFTBUFBBS7)

Offset

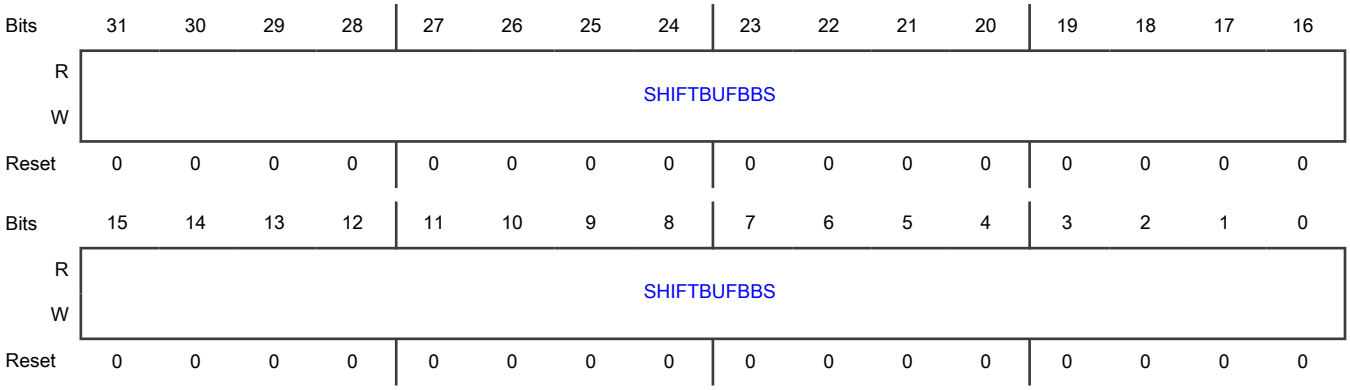
For a = 0 to 7:

Register	Offset
SHIFTBUFBBSa	380h + (a × 4h)

Function

Contains SHIFTBUF register data, but is bit swapped within each byte.

Diagram



Fields

Field	Function
31-0 SHIFTBUFBBS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are bit swapped within each byte. Reads return { SHIFTBUF[24:31], SHIFTBUF[16:23], SHIFTBUF[8:15], SHIFTBUF[0:7] }.

69.6.1.33 Timer Control N Register (TIMCTL0 - TIMCTL7)

Offset

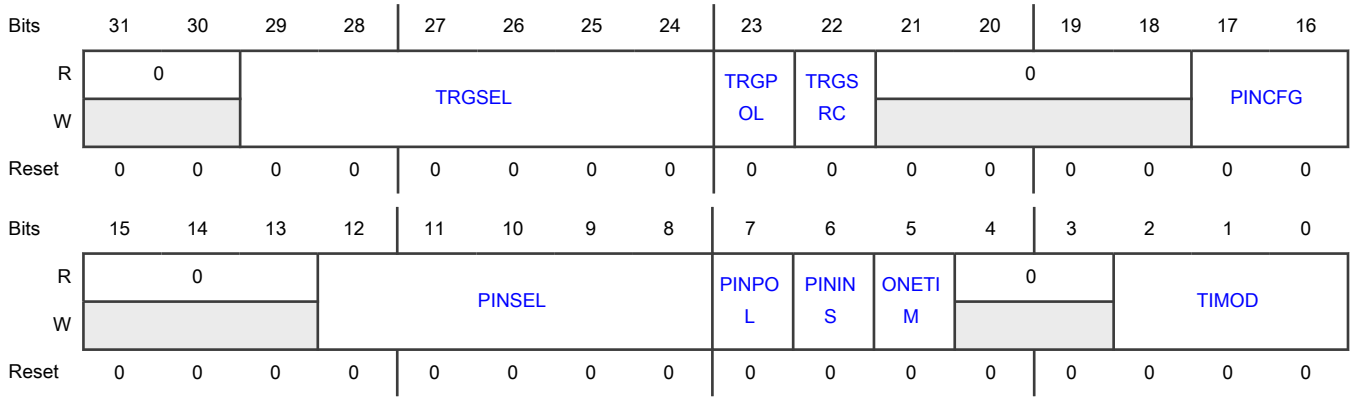
For a = 0 to 7:

Register	Offset
TIMCTLa	400h + (a × 4h)

Function

Timer Control for Timer N.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 TRGSEL	Trigger Select The valid values for TRGSEL depends on the FLEXIO_PARAM register. • When TRGSRC = 1, the valid values for N depends on PIN, TIMER, SHIFTER fields in the FLEXIO_PARAM register. • When TRGSRC = 0, the valid values for N depends on TRIGGER field in FLEXIO_PARAM register. Refer to the chip-specific FlexIO information for external trigger selection. **NOTE** For a pin, N=0 to 31. For a Shifter, N=0 to 7. For a Timer, N=0 to 7. When TRGSRC = 0 the trigger selection is configured as follows: • N - External trigger N input When TRGSRC = 1 the internal trigger can be configured to select an input pin as follows: • 2*N - Pin N input When TRGSRC = 1 the internal trigger can be configured to select a shifter/timer signal as follows: • 4*N+1 - Shifter N status flag • 4*N+3 - Timer N trigger output In other words, the expanded internal trigger selection (TRGSRC = 1) is as follows: • 0000 = Pin 0 • 0001 = Shifter 0 Flag • 0010 = Pin 1 • 0011 = Timer 0 Trigger

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0100 = Pin 2 • 0101 = Shifter 1 Flag • 0110 = Pin 3 • 0111 = Timer 1 Trigger • ... • This continues up to the Pin 31, Shifter 7 and Timer 7.
23 TRGPOL	Trigger Polarity Determines if the trigger is active high or active low. 0b - Trigger active high 1b - Trigger active low
22 TRGSRC	Trigger Source Determines if the trigger source is external or internal. 0b - External trigger selected 1b - Internal trigger selected
21-18 —	Reserved
17-16 PINCFG	Timer Pin Configuration Configures the direction of the Timer pin. For pins configured as an output (PINCFG=11), this field takes effect when the register is written. 00b - Timer pin output disabled 01b - Timer pin open drain or bidirectional output enable 10b - Timer pin bidirectional output data 11b - Timer pin output
15-13 —	Reserved
12-8 PINSEL	Timer Pin Select Selects which pin is used by the Timer input or output. PINSEL=i selects the FXIO_Di pin. For pins configured as an output (PINCFG=11), this field takes effect when the register is written.
7 PINPOL	Timer Pin Polarity For pins configured as an output (PINCFG=11), this field takes effect when the register is written. 0b - Pin is active high 1b - Pin is active low

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 PININS	Timer Pin Input Select When set, the timer input pin is a different pin from the timer output pin. PINSEL must select an even numbered pin when this bit is set, so the output pin is even numbered and input pin is odd numbered. 0b - Timer pin input and output are selected by PINSEL. 1b - Timer pin input is selected by PINSEL+1, timer pin output remains selected by PINSEL.
5 ONETIM	Timer One Time Operation When set, configures the timer to perform a single enable/disable iteration, after which software must clear the timer status flag for the timer to be enabled again. 0b - The timer enable event is generated as normal. 1b - The timer enable event is blocked unless timer status flag is clear.
4-3 —	Reserved
2-0 TIMOD	Timer Mode In 8-bit baud counter mode, the lower 8-bits of the counter and compare register are used to configure the baud rate of the timer shift clock, and the upper 8-bits are used to configure the shifter bit count. In 8-bit PWM high mode, the lower 8-bits of the counter and compare register are used to configure the high period of the timer shift clock, and the upper 8-bits are used to configure the low period of the timer shift clock. The shifter bit count is configured using another timer or external signal. In 16-bit counter mode, the full 16-bits of the counter and compare register are used to configure either the baud rate of the shift clock or the shifter bit count. In 16-bit counter disable mode, the full 16-bits of the counter and compare register are used to configure either the baud rate of the shift clock or the shifter bit count. In 8-bit word counter mode, the lower 8-bits of the counter and compare register are used to configure the shifter bit count, and the upper 8-bits are used to configure the shifter word count. In 8-bit PWM low mode, the lower 8-bits of the counter and compare register are used to configure the low period of the timer shift clock and the upper 8-bits are used to configure the high period of the timer shift clock. The shifter bit count is configured using another timer or external signal. In 16-bit input capture mode, the inverted value of 16-bit counter is latched into the compare register when a timer counter disable condition is detected (as configured by TIMDIS). This also sets the timer status flag. The timer counter is immediately restarted from 0xFFFF 000b - Timer Disabled. 001b - Dual 8-bit counters baud mode. 010b - Dual 8-bit counters PWM high mode. 011b - Single 16-bit counter mode. 100b - Single 16-bit counter disable mode. 101b - Dual 8-bit counters word mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - Dual 8-bit counters PWM low mode. 111b - Single 16-bit input capture mode.

69.6.1.34 Timer Configuration N Register (TIMCFG0 - TIMCFG7)

Offset

For a = 0 to 7:

Register	Offset
TIMCFGa	480h + (a × 4h)

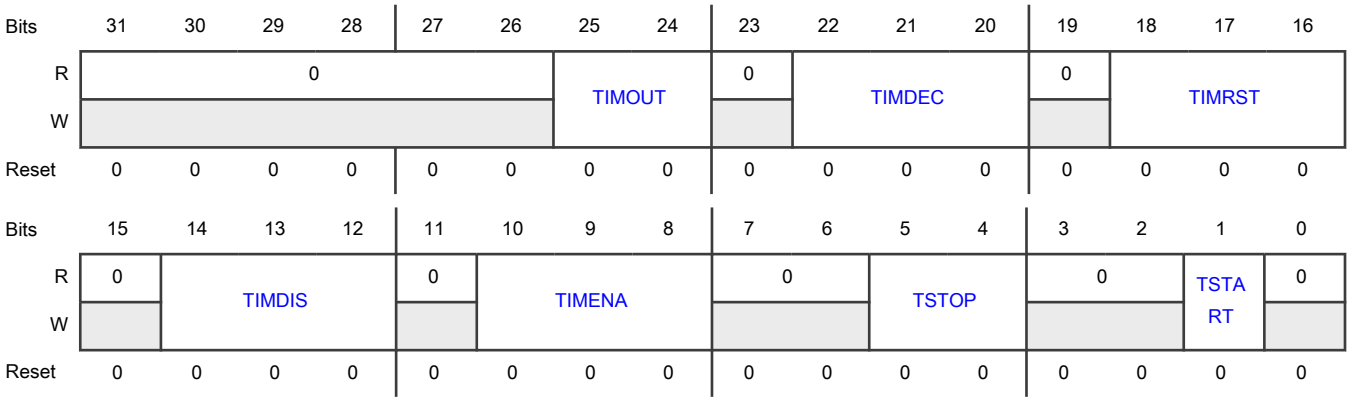
Function

The options to enable or disable the timer using the Timer N-1 enable or disable are reserved when N is evenly divisible by 4 (eg: Timer 0).

NOTE

The pin and trigger level/edges specified below refer to the signal state after being modified by the PINPOL or TRGPOL setting in the TIMCTL register. For example, "Trigger low" means a trigger is actually at logic level 1 if the TRGPOL is set to 1 (active low).

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24	Timer Output

Table continues on the next page...

Table continued from the previous page...

Field	Function
TIMOUT	Configures the initial state of the Timer Output and whether it is affected by the Timer reset. 00b - Timer output is logic one when enabled and is not affected by timer reset 01b - Timer output is logic zero when enabled and is not affected by timer reset 10b - Timer output is logic one when enabled and on timer reset 11b - Timer output is logic zero when enabled and on timer reset
23 —	Reserved
22-20 TIMDEC	Timer Decrement Configures the source of the Timer decrement and the source of the Shift clock. 000b - Decrement counter on FlexIO clock, Shift clock equals Timer output. 001b - Decrement counter on Trigger input (both edges), Shift clock equals Timer output. 010b - Decrement counter on Pin input (both edges), Shift clock equals Pin input. 011b - Decrement counter on Trigger input (both edges), Shift clock equals Trigger input. 100b - Decrement counter on FlexIO clock divided by 16, Shift clock equals Timer output. 101b - Decrement counter on FlexIO clock divided by 256, Shift clock equals Timer output. 110b - Decrement counter on Pin input (rising edge), Shift clock equals Pin input. 111b - Decrement counter on Trigger input (rising edge), Shift clock equals Trigger input.
19 —	Reserved
18-16 TIMRST	Timer Reset Configures the condition that causes the timer counter (and optionally the timer output) to be reset. In 8-bit counter mode, the timer reset only resets the lower 8-bits that configure the baud rate. In all other modes, the timer reset resets the full 16-bits of the counter. 000b - Timer never reset 001b - Timer reset on Timer Output high. 010b - Timer reset on Timer Pin equal to Timer Output 011b - Timer reset on Timer Trigger equal to Timer Output 100b - Timer reset on Timer Pin rising edge 101b - Reserved 110b - Timer reset on Trigger rising edge 111b - Timer reset on Trigger rising or falling edge
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-12 TIMDIS	Timer Disable Configures the condition that causes the Timer to be disabled and stop decrementing. 000b - Timer never disabled 001b - Timer disabled on Timer N-1 disable 010b - Timer disabled on Timer compare (upper 8-bits match and decrement) 011b - Timer disabled on Timer compare (upper 8-bits match and decrement) and Trigger Low 100b - Timer disabled on Pin rising or falling edge 101b - Timer disabled on Pin rising or falling edge provided Trigger is high 110b - Timer disabled on Trigger falling edge 111b - Reserved
11 —	Reserved
10-8 TIMENA	Timer Enable Configures the condition that causes the Timer to be enabled and start decrementing. 000b - Timer always enabled 001b - Timer enabled on Timer N-1 enable 010b - Timer enabled on Trigger high 011b - Timer enabled on Trigger high and Pin high 100b - Timer enabled on Pin rising edge 101b - Timer enabled on Pin rising edge and Trigger high 110b - Timer enabled on Trigger rising edge 111b - Timer enabled on Trigger rising or falling edge
7-6 —	Reserved
5-4 TSTOP	Timer Stop Bit The stop bit can be added on a timer compare (between each word) or on a timer disable. When stop bit is enabled, configured shifters output the contents of the stop bit when the timer is disabled. When stop bit is enabled on timer disable, the timer remains disabled until the next rising edge of the shift clock. If configured for both timer compare and timer disable, only one stop bit is inserted on timer disable. 00b - Stop bit disabled 01b - Stop bit is enabled on timer compare 10b - Stop bit is enabled on timer disable 11b - Stop bit is enabled on timer compare and timer disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-2 —	Reserved
1 TSTART	Timer Start Bit When start bit is enabled, configured shifters outputs the contents of the start bit when the timer is enabled and the timer counter reloads from the compare register on the first rising edge of the shift clock. 0b - Start bit disabled 1b - Start bit enabled
0 —	Reserved

69.6.1.35 Timer Compare N Register (TIMCMP0 - TIMCMP7)

Offset

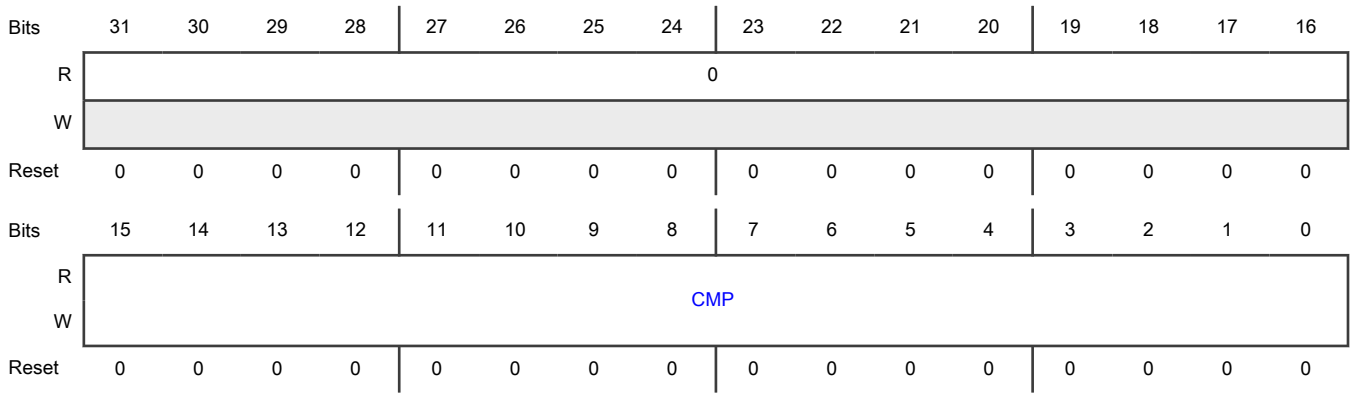
For a = 0 to 7:

Register	Offset
TIMCMPa	500h + (a × 4h)

Function

Contains the timer compare value.

Diagram



Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-0 CMP	Timer Compare Value The timer compare value is loaded into the timer counter when the timer is first enabled, when the timer is reset and when the timer decrements down to zero. In 8-bit baud counter mode, the lower 8-bits configure the baud rate divider equal to (CMP[7:0] + 1) * 2. The upper 8-bits configure the number of bits in each word equal to (CMP[15:8] + 1) / 2. In 8-bit PWM high mode, the lower 8-bits configure the high period of the output to (CMP[7:0] + 1) and the upper 8-bits configure the low period of the output to (CMP[15:8] + 1). In 16-bit counter mode, the compare value can be used to generate the baud rate divider (if shift clock source is timer output) to equal (CMP[15:0] + 1) * 2. When the shift clock source is a pin or trigger input, the compare register is used to set the number of bits in each word equal to (CMP[15:0] + 1) / 2. In 16-bit counter disable mode, the compare value can be used to generate the baud rate divider (if shift clock source is timer output) to equal (CMP[15:0] + 1) * 2. When the shift clock source is a pin or trigger input, the compare register is used to set the number of bits in each word equal to (CMP[15:0] + 1) / 2. In 8-bit word counter mode, the lower 8-bits configure the number of bits in each word equal to (CMP[7:0] + 1) / 2. The upper 8-bits configure the number of words to transfer equal to (CMP[15:8] + 1) / 2. In 8-bit PWM low mode, the lower 8-bits configure the low period of the output to (CMP[7:0] + 1) and the upper 8-bits configure the high period of the output to (CMP[15:8] + 1). In 16-bit input capture mode, the compare register is updated with the inverse of the timer counter value whenever the timer status flag is set. The timer compare register should only be read when the timer status flag is set.

69.6.1.36 Shifter Buffer N Nibble Byte Swapped Register (SHIFTBUFNBS0 - SHIFTBUFNBS7)

Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFNBSa	680h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register are nibble swapped within each byte.

Diagram

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

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

Fields

Field	Function
31-0 SHIFTBUFNBS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are nibble swapped within each byte. Reads return { SHIFTBUF[27:24], SHIFTBUF[31:28], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[3:0], SHIFTBUF[7:4] }.

69.6.1.37 Shifter Buffer N Half Word Swapped Register (SHIFTBUFHWS0 - SHIFTBUFHWS7)

Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFHWSa	700h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register are half word swapped.

Diagram

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

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

Fields

Field	Function
31-0 SHIFTBUFHWS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are half word swapped. Reads return { SHIFTBUF[15:0], SHIFTBUF[31:16] }.

69.6.1.38 Shifter Buffer N Nibble Swapped Register (SHIFTBUFNIS0 - SHIFTBUFNIS7)

Offset

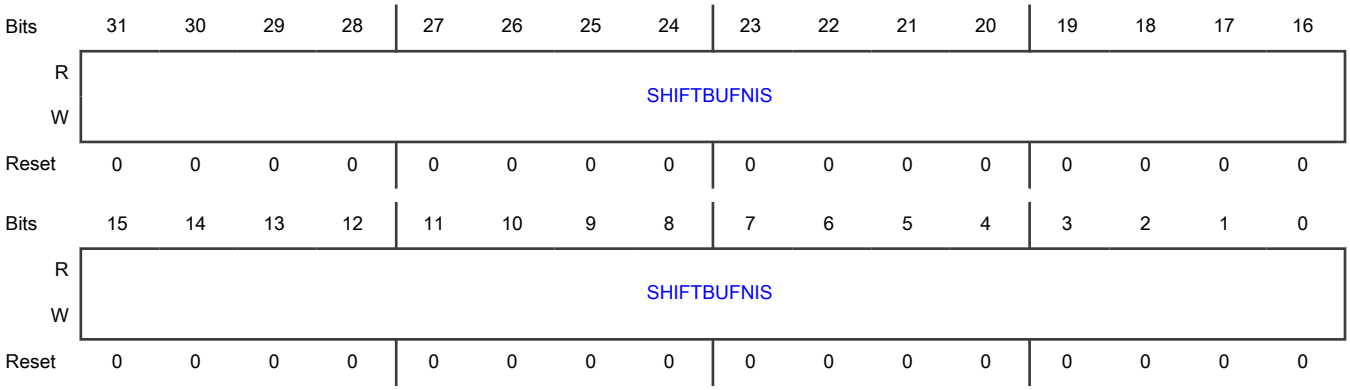
For a = 0 to 7:

Register	Offset
SHIFTBUFNISa	780h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register are nibble swapped.

Diagram



Fields

Field	Function
31-0 SHIFTBUFNIS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are nibble swapped. Reads return { SHIFTBUF[3:0], SHIFTBUF[7:4], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[27:24], SHIFTBUF[31:28] }.

69.6.1.39 Shifter Buffer N Odd Even Swapped Register (SHIFTBUFOES0 - SHIFTBUFOES7)

Offset

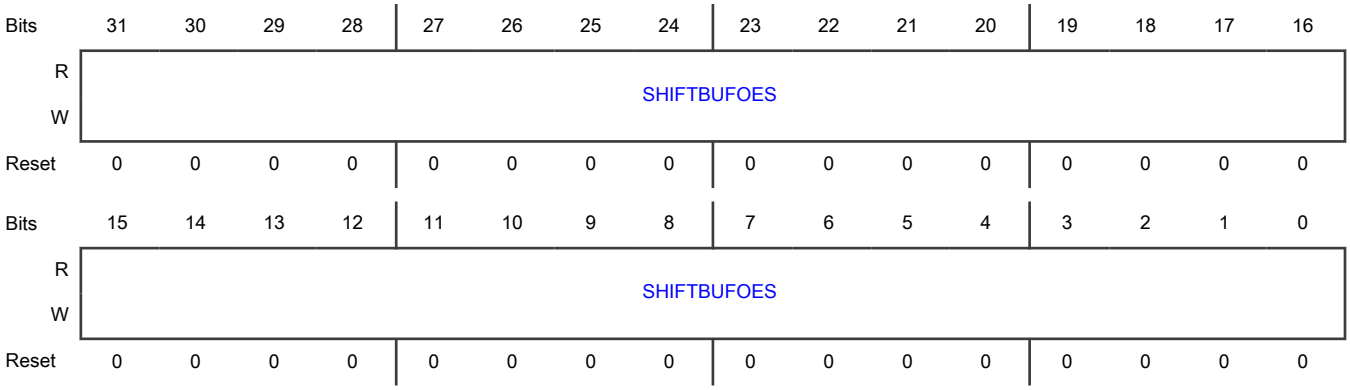
For a = 0 to 7:

Register	Offset
SHIFTBUFOESa	800h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register have odd and even bits partioned separately.

Diagram



Fields

Field	Function
31-0 SHIFTBUFOES	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register have the odd and even bits partitioned separately. Only 32-bit accesses are supported to this register. Reads return { SHIFTBUF[31], SHIFTBUF[29], SHIFTBUF[27],SHIFTBUF[25],SHIFTBUF[23],SHIFTBUF[21],SHIFTBUF[19],SHIFTBUF[17],SHIFTBUF[15],SHIFTBUF[13],SHIFTBUF[11],SHIFTBUF[9], SHIFTBUF[7], SHIFTBUF[5], SHIFTBUF[3], SHIFTBUF[1], SHIFTBUF[30], SHIFTBUF[28], SHIFTBUF[26], SHIFTBUF[24], SHIFTBUF[22], SHIFTBUF[20], SHIFTBUF[18], SHIFTBUF[16], SHIFTBUF[14], SHIFTBUF[12], SHIFTBUF[10], SHIFTBUF[8], SHIFTBUF[6], SHIFTBUF[4], SHIFTBUF[2], SHIFTBUF[0] }.

69.6.1.40 Shifter Buffer N Even Odd Swapped Register (SHIFTBUFEOS0 - SHIFTBUFEOS7)

Offset

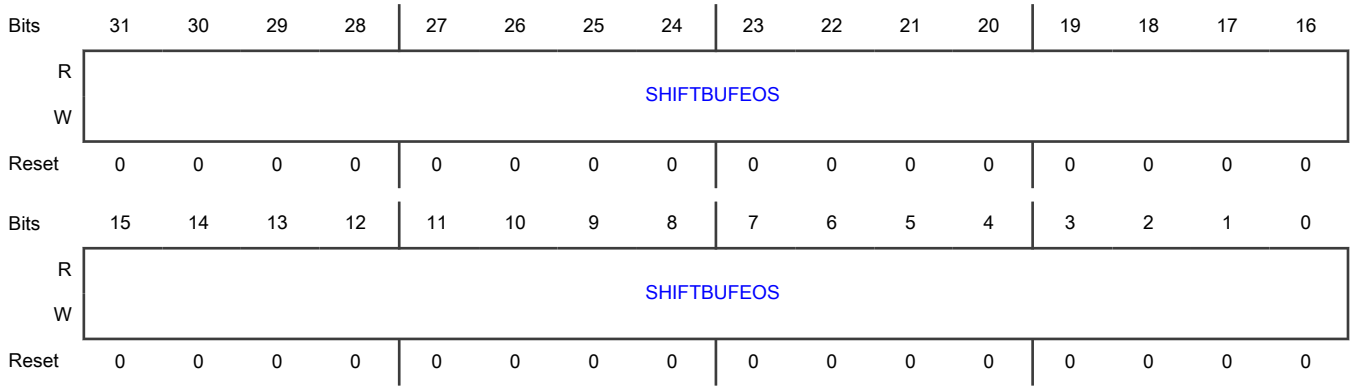
For a = 0 to 7:

Register	Offset
SHIFTBUFEOSa	880h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register have even and odd bits partioned separately.

Diagram



Fields

Field	Function
31-0 SHIFTBUFEOS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register have the even and odd bits partitioned separately. Only 32-bit accesses are supported to this register. Reads return { SHIFTBUF[30], SHIFTBUF[28], SHIFTBUF[26], SHIFTBUF[24], SHIFTBUF[22], SHIFTBUF[20], SHIFTBUF[18], SHIFTBUF[16], SHIFTBUF[14], SHIFTBUF[12], SHIFTBUF[10], SHIFTBUF[8], SHIFTBUF[6], SHIFTBUF[4], SHIFTBUF[2], SHIFTBUF[0], SHIFTBUF[31], SHIFTBUF[29], SHIFTBUF[27],SHIFTBUF[25],SHIFTBUF[23],SHIFTBUF[21],SHIFTBUF[19],SHIFTBUF[17],SHIFTBUF[15],SHIFTBUF[13],SHIFTBUF[11],SHIFTBUF[9], SHIFTBUF[7], SHIFTBUF[5], SHIFTBUF[3], SHIFTBUF[1] }.

69.6.1.41 Shifter Buffer N Halfword Byte Swapped Register (SHIFTBUFHBS0 - SHIFTBUFHBS7)

Offset

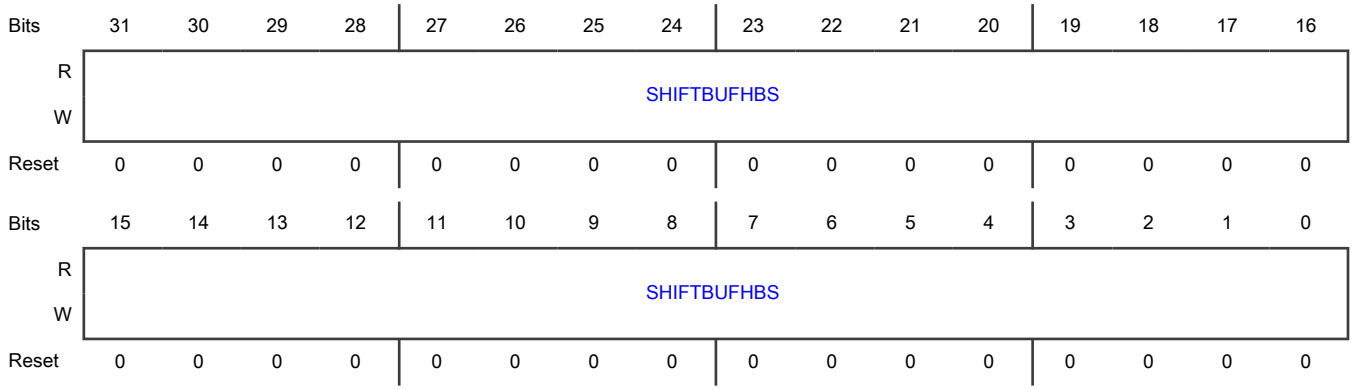
For a = 0 to 7:

Register	Offset
SHIFTBUFHBSa	900h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register are halfword byte swapped.

Diagram



Fields

Field	Function
31-0 SHIFTBUFHBS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are halfword byte swapped. Reads return { SHIFTBUF[23:16], SHIFTBUF[31:24], SHIFTBUF[7:0], SHIFTBUF[15:8] }.

69.7 Glossary

Baud rate Number of bits transmitted or received per second by the LPUART

Break character Break character is generated when the transmitter is holding the data line at the space level for at least 1 character time

Idle character Idle character is generated when the transmitter is holding the data line at logic 1 for at least 1 character time

PWM Pulse Width Modulation

Chapter 70
CAN (FlexCAN)

70.1 Chip-specific FlexCAN information

70.1.1 FlexCAN instances and configuration

This chip contains up to 8 instances of FlexCAN:

- FlexCAN_0
- FlexCAN_1
- FlexCAN_2
- FlexCAN_3
- FlexCAN_4
- FlexCAN_5
- FlexCAN_6
- FlexCAN_7

NOTE

The availability of 'Interrupt Masks 3 Register (IMASK3)' and 'Interrupt Flags 3 Register (IFLAG3)' registers for an instance depends upon the number of message buffers supported. See Table 413 for details.

Table 413. FlexCAN configuration and instances

Chip	Instances	Flexible Data Rate (CAN FD) protocol specification and CAN protocol.	Number of message buffers (assuming 8B payload)	Rx FIFO ID Filtering	ECC	External tick time	DMA	Enhanced RX FIFO feature (size)
MWCT2D17S	FlexCAN_0	Yes	96	Yes	Yes	Yes (PIT_RTI_2)	Yes	Yes (20)
	FlexCAN_1	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_2	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_3	Yes	32	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_4	Yes	32	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_5	Yes	32	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
MWCT2D16S	FlexCAN_0	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	Yes (20)

Table continues on the next page...

Table 413. FlexCAN configuration and instances (continued)

Chip	Instances	Flexible Data Rate (CAN FD) protocol specification and CAN protocol.	Number of message buffers (assuming 8B payload)	Rx FIFO ID Filtering	ECC	External tick time	DMA	Enhanced RX FIFO feature (size)
	FlexCAN_1	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_2	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_3	Yes	32	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
MWCT2016SG	FlexCAN_0	Yes	64	Yes	Yes	Yes (PIT_RTI_1)	Yes	Yes (20)
	FlexCAN_1	Yes	64	Yes	Yes	Yes (PIT_RTI_1)	Yes	No
	FlexCAN_2	Yes	64	Yes	Yes	Yes (PIT_RTI_1)	Yes	No
	FlexCAN_3	Yes	32	Yes	Yes	Yes (PIT_RTI_1)	Yes	No
	FlexCAN_4	Yes	32	Yes	Yes	Yes (PIT_RTI_1)	Yes	No
	FlexCAN_5	Yes	32	Yes	Yes	Yes (PIT_RTI_1)	Yes	No
MWCT2015S	FlexCAN_0	Yes	64	Yes	Yes	Yes (PIT_RTI_1)	Yes	Yes (20)
	FlexCAN_1	Yes	64	Yes	Yes	Yes (PIT_RTI_1)	Yes	No
	FlexCAN_2	Yes	64	Yes	Yes	Yes (PIT_RTI_1)	Yes	No

NOTE

See attached interrupt map file for the vector numbers for the CAN instances and their corresponding interrupt sources.

70.1.2 FlexCAN error injection

Error injection address mapping

Use the following table to convert from the memory map address to the location in the physical FlexCAN RAM (where pairs of values are provided, the first is the address for MCR[FDEN] negated, the second is for MCR[FDEN] asserted):

RAM contents	Memory map	Injection address						
—	—	FlexCAN0	FlexCAN0	FlexCAN0/1/2	FlexCAN1/2	FlexCAN3	FlexCAN4/5	FlexCAN3/4/5/6/7
					MWCT2xxxS			
		MWCT2D17S				MWCT2D17S	MWCT2D17S	
			MWCT2D16S					
			MWCT2016S				MWCT2016S	
			MWCT2015S					
						MWCT2D16S		
						MWCT2016S		
FlexCAN Registers	—	Not mapped	Not mapped	Not mapped	Not mapped	Not mapped	Not mapped	Not mapped
MBs	0080h	0000h	0000h	0000h	0000h	0000h	0000h	0000h
RXIMRs	0880h	600h	400h	600h	400h	200h	200h	400h
RXIFR_0	0A80h	600h	500h	600h	500h	280h	280h	500h
RXIFR_1	0A84h	784h	504h	784h	504h	284h	284h	504h
RXIFR_2	0A88h	788h	508h	788h	508h	288h	288h	508h
RXIFR_3	0A8Ch	78Ch	50Ch	78Ch	50Ch	28Ch	28Ch	50Ch
RXIFR_4	0A90h	790h	510h	790h	510h	290h	290h	510h
RXIFR_5	0A94h	794h	514h	794h	514h	294h	294h	514h
Reserved	0A98h	—	—	—	—	—	—	—
RXMGMASK	0AA0h	7A0h	520h	7A0h	520h	2A0h	2A0h	520h
RXFGMASK	0AA4h	7A4h	524h	7A4h	524h	2A4h	2A4h	524h
RX14MASK	0AA8h	7A8h	528h	7A8h	528h	2A8h	2A8h	528h
RX15MASK	0AACh	7ACh	52Ch	7ACh	52Ch	2ACh	2Ach	52Ch
Tx_SMB	0AB0h/0F28h	7B0h	530h	7B0h	530h	2B0h	2B0h	530h
Rx_SMB0	0AC0h/0F70h	7C0h/7F8h	540h/578h	7C0h/7F8h	540h/578h	2C0h/2F8h	2C0h/2F8h	540h/578h
Rx_SMB1	0AD0h/0FB8h	7D0h/840h	550h/5C0h	7D0h/840h	550h/5C0h	2D0h/340h	2D0h/340h	550h/5C0h

Table continues on the next page...

Table continued from the previous page...

RAM contents	Memory map	Injection address						
Rx_SMB0_TIME_STAMP	0C20h	888h	608h	888h	608h	388h	388h	608h
Rx_SMB1_TIME_STAMP	0C24h	88Ch	60Ch	88Ch	60Ch	38Ch	38Ch	60Ch
HR_TIME_STAMP	0C30h	890h	610h	890h	610h	390h	390h	610h
Enhanced RX FIFO	2000h	A10h	710h	A10h	—	—	—	—
ERFFEL	3000h	1050h	D50h	1050h	—	—	—	—

FlexCAN memory initialization

All FlexCAN memory must be initialized before starting its operation in order to have the parity bits in memory properly updated. CTRL2[WRMFRZ] grants write access to all memory positions that require initialization.

These locations are as listed in this table from 080h–ADFh and from C20h–31FFh.:

Chip	Instance	Offset address ranges to be initialized
	FlexCAN3–FlexCAN7	080h–ADFh; C20h–FFFh
MWCT2D17S	FlexCAN0	080h–ADFh; C20h–31FFh
	FlexCAN1–FlexCAN5	080h–ADFh; C20h–FFFh
MWCT2D16S	FlexCAN0	080h–ADFh; C20h–31FFh
	FlexCAN1–FlexCAN3	080h–ADFh; C20h–FFFh
MWCT2016S	FlexCAN0	080h–DFh; C20h–31FFh
	FlexCAN1–FlexCAN5	080h–ADFh; C20h–FFFh
MWCT2015S	FlexCAN0	080h–ADFh; C20h–31FFh
	FlexCAN1–FlexCAN2	080h–ADFh; C20h–FFFh

NOTE

The reset value of CAN0_CTRL2 register for MWCT2016S and MWCT2D16S is 0080_0000h.

70.1.3 Reset value of MDIS bit

The CAN_MCR[MDIS] bit is set to 1 when the module is reset. Therefore, FlexCAN module is disabled following a reset. For details of CAN_MCR[MDIS] register refer to "CAN register description" section.

70.1.4 CAN Timestamp Implementation

FlexCAN uses Ethernet Media Access Controller (EMAC) and System Timer Module (STM_0) counters as time source. For details see 'FlexCAN timestamp implementation' section in Clocking chapter.

70.1.5 CAN clock setup

For clock setup, see Clocking chapter.

70.2 Overview

The FlexCAN module is a communication controller implementing the CAN protocol according to the ISO 11898-1 standard and CAN 2.0 B protocol specifications.

The CAN protocol was primarily designed to be used as a vehicle serial data bus, meeting the specific real-time processing and reliable operation requirements in the EMI environment of a vehicle. The FlexCAN module is a full implementation of the CAN protocol specification, the CAN with Flexible Data rate (CAN FD) protocol, and the CAN 2.0 version B protocol, which supports both standard and extended message frames and long payloads.

NOTE

Legacy Rx FIFO cannot be used in FD mode.

NOTE

In CAN FD mode, the Enhanced Rx FIFO feature should be used instead of Legacy Rx FIFO.

70.2.1 Block diagram

A general block diagram is shown in Figure 372, which describes the main subblocks implemented in the FlexCAN module.

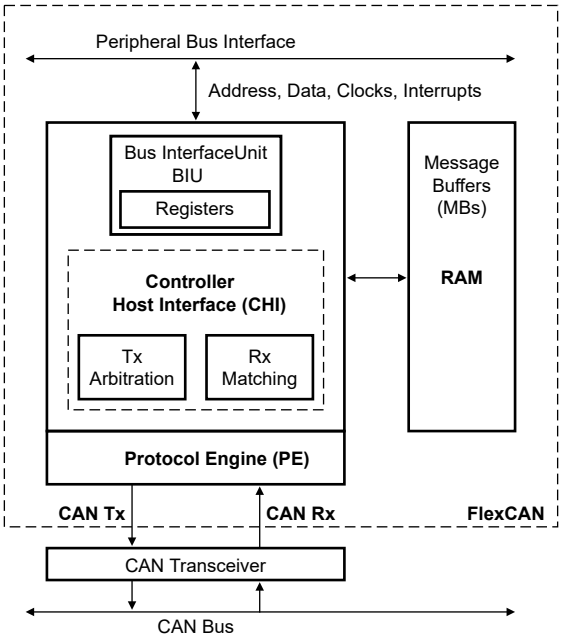


Figure 372. FlexCAN block diagram

The Protocol Engine (PE) submodule manages the serial communication on the CAN bus:

- Requesting RAM access for receiving and transmitting message frames
- Validating received messages
- Performing error handling
- Detecting CAN FD messages

The Controller Host Interface (CHI) submodule manages message buffer selection for reception and transmission, taking care of arbitration and ID matching algorithms for both CAN FD and non-CAN FD message formats.

The Bus Interface Unit (BIU) submodule controls access to and from the internal interface bus, in order to establish connection to the CPU and to other blocks. Clocks, address and data buses, interrupt outputs, DMA and test signals are accessed through the BIU.

70.2.2 Features

The FlexCAN module includes these distinctive features:

- Full implementation of the CAN with Flexible Data Rate (CAN FD) protocol specification and CAN protocol specification, Version 2.0 B
  - Standard data frames
  - Extended data frames
  - Zero to sixty four bytes data length
  - Programmable bit rate (see the chip-specific FlexCAN information for the specific maximum rate configuration)
  - Content-related addressing
- Compliant with the ISO 11898-1 standard
- Flexible mailboxes configurable to store 0 to 8, 16, 32, or 64 bytes data length
- Each mailbox configurable as receive or transmit, all supporting standard and extended messages
- Individual Rx Mask registers per mailbox
- Full-featured legacy Rx FIFO with storage capacity for up to six frames and automatic internal pointer handling with DMA support
- Full-featured enhanced Rx FIFO with storage capacity for up to 20 CAN FD frames and automatic internal pointer handling with DMA support
- Transmission abort capability
- Flexible message buffers (MBs), totaling 96 message buffers of 8 bytes data length each, configurable as Rx or Tx
- RAM not used by reception or transmission structures can be used as general purpose RAM space
- Listen-Only mode capability
- Programmable Loop-Back mode supporting self-test operation
- Programmable transmission priority scheme: lowest ID, lowest buffer number, or highest priority
- Time stamp based on 32-bit free running timer, with an optional external time tick
- Global network time, synchronized by a specific message
- Maskable interrupts
- Independence from the transmission medium (an external transceiver is assumed)
- Short latency time due to an arbitration scheme for high-priority messages
- Low power mode
- Transceiver delay compensation feature when transmitting CAN FD messages at faster data rates
- Remote request frames may be managed automatically or by software
- CAN bit time settings and configuration bits can only be written in Freeze mode
- Tx mailbox status (Lowest priority buffer or empty buffer)
- Identifier Acceptance Filter Hit Indicator (IDHIT) register for received frames
- SYNCH bit available in Error in Status 1 register to indicate that the module is synchronous with CAN bus
- CRC status for transmitted message

- Legacy Rx FIFO Global Mask register
- Selectable priority between mailboxes and Rx FIFO during matching process
- Powerful legacy Rx FIFO ID filtering, capable of matching incoming IDs against either 128 extended, 256 standard, or 512 partial (8 bit) IDs, with up to 32 individual masking capability
- Powerful Enhanced Rx FIFO ID filtering, capable of matching incoming IDs against either 64 extended or 128 standard ID filter elements with three filtering schemes: mask + filter, range, and two filters without mask.
- 100% backward compatibility with previous FlexCAN version
- Supports detection and correction of errors in memory read accesses

  Each byte of FlexCAN memory is associated to 5 parity bits, and the error correction mechanism ensures that in this 13-bit word, errors in one bit can be corrected (correctable errors) and errors in 2 bits can be detected but not corrected (non-correctable errors).

70.3 Functional description

The FlexCAN module is a CAN protocol engine with a very flexible mailbox system for transmitting and receiving CAN frames. The mailbox system is composed of a set of message buffers (MB) that store configuration and control data, time stamp, message ID, and data (see Message buffer structure). The memory corresponding to the first 38 MBs can be configured to support a Legacy FIFO reception scheme with a powerful ID filtering mechanism, capable of checking incoming frames against a table of IDs (up to 128 extended IDs or 256 standard IDs or 512 8-bit ID slices), with individual mask register for up to 32 ID filter table elements.

For Classical CAN frames, simultaneous reception through Legacy FIFO and mailbox is supported. For CAN FD frames, reception is supported through mailboxes and Enhanced Rx FIFO. For mailbox reception, a matching algorithm makes it possible to store received frames only into MBs that have the same ID programmed in the ID field. A masking scheme makes it possible to match the ID programmed on the MB with a range of IDs on received CAN frames. For transmission, an arbitration algorithm decides the prioritization of MBs to be transmitted based on the message ID (optionally augmented by 3 local priority bits) or the MB ordering.

Before proceeding with the functional description, an important concept must be explained. A message buffer is said to be active at a given time if it can participate in both the matching and arbitration processes. An Rx MB with a 0b code is inactive (see Table 446). Similarly, a Tx MB with a 1000b or 1001b code is also inactive (see Table 447).

The FlexCAN module is also able to receive and transmit messages in CAN FD format. The message buffers are sized to adequately store the quantity of data bytes selected by the FDCTRL[MBDSRn] fields. The quantity of FD MBs available for a given quantity of data bytes is described in the FDCTRL register. See also FlexCAN memory partition for CAN FD.

70.3.1 Modes of operation

The FlexCAN module has these functional modes:

- Normal mode (User or Supervisor):

  In Normal mode, the module operates receiving and/or transmitting message frames, errors are managed normally, and all CAN Protocol functions are enabled. User and Supervisor modes differ in the access to some restricted control registers.

- Freeze mode:

  Freeze mode is enabled when MCR[FRZ] is asserted. If enabled, Freeze mode is entered when MCR[HALT] is set or when is requested at chip level and MCR[FRZ_ACK] is asserted by the FlexCAN. In this mode, no transmission or reception of frames is done and synchronicity to the CAN bus is lost. See Freeze mode for more information.

- Loop-Back mode:

  The module enters this mode when CTRL1[LPB] is asserted. In this mode, FlexCAN performs an internal loop back that can be used for self-test operation. The bit stream output of the transmitter is internally fed back to the receiver input. The Rx CAN input pin is ignored and the Tx CAN output goes to the recessive state (logic '1'). FlexCAN behaves as it normally does when transmitting and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field to ensure proper reception of its own message. Both transmit and receive interrupts are generated.

- Listen-Only mode:

  The module enters this mode when CTRL1[LOM] is asserted. In this mode, transmission is disabled, all error counters are frozen, and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station will be received. If FlexCAN detects a message that has not been acknowledged, it will flag a BIT0 error (without changing the REC), as if it was trying to acknowledge the message.

- CAN FD Active mode:

  In this mode, FlexCAN is capable of transmitting and receiving all messages formatted according to the CAN FD Standard (2.0) and 2.0B Protocol in an interleaved fashion. The CPU can set the FlexCAN into CAN FD Active mode by configuring MCR[FDEN] in Freeze mode.

  It is important to know which features are available in CAN FD active mode. This table lists the differences between FD and classical modes.

Table 414. Differences between CAN classical and CAN FD

Feature	CAN classical	CAN FD
Legacy Rx FIFO DMA	Yes	No
Legacy Rx FIFO	Yes	No
Enhanced Rx FIFO DMA	Yes	Yes
Enhanced Rx FIFO	Yes	Yes

For low-power operation, the FlexCAN module has:

- Module Disable mode:

  This low-power mode is entered when MCR[MDIS] is asserted by the CPU and MCR[LPM_ACK] is asserted by FlexCAN. When disabled, the module issues a request to disable the clocks to the CAN Protocol Engine and Controller Host Interface submodules. Exit from this mode is done by negating MCR[MDIS]. See Module Disable mode for more information.

70.3.1.1 Modes of operation details

The FlexCAN module has functional modes and low-power modes. See Modes of operation for an introductory description of all the modes of operation. The following sub-sections contain functional details on Freeze mode and the low-power modes.

CAUTION

"Permanent Dominant" failure on CAN Bus line is not supported by FlexCAN. If a Low-Power request or Freeze mode request is done during a "Permanent Dominant", the corresponding acknowledge can never be asserted.

70.3.1.1.1 Freeze mode

This mode is requested either by the CPU through the assertion of MCR[HALT] or when the chip is put into Debug mode. In both cases it is also necessary that MCR[FRZ] be asserted and the module not be in a low-power mode. This mode is also requested by FlexCAN through the automatic assertion of both MCR[HALT] and MCR[FRZ] when MECR[NCEFAFRZ] is set and a non-correctable error is detected in a memory read access performed by FlexCAN internal processes (see Response to errors).

The acknowledgement is obtained through the assertion by the FlexCAN of MECR[FRZ_ACK]. The CPU must only consider the FlexCAN in Freeze mode when both request and acknowledgement conditions are satisfied.

When Freeze mode is requested, FlexCAN does the following:

- Waits to be in either Intermission, Passive Error, Bus Off, or Idle state.
- Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. A pending move-in does not prevent entering Freeze mode.
- Ignores the Rx input pin and drives the Tx pin as recessive.
- Stops the prescaler, thus halting all CAN protocol activities.

- Grants write access to the Error Counters register, which is read-only in other modes.
- Sets MCR[NOTRDY] and MCR[FRZACK].

After requesting Freeze mode, the user must wait for MCR[FRZ_ACK] to be asserted before executing any other action; otherwise FlexCAN may operate in an unpredictable way. In Freeze mode, all memory-mapped registers are accessible.

Exiting Freeze mode is done in one of the following ways:

- CPU negates MCR[FRZ].
- The chip is removed from Debug mode and/or the HALT bit is negated.

MCR[FRZ_ACK] is negated after the protocol engine recognizes the negation of the freeze request. When out of Freeze mode, FlexCAN tries to resynchronize to the CAN bus by waiting for 11 consecutive recessive bits.

70.3.1.1.2 Module Disable mode

This low-power mode is normally used to temporarily disable a complete FlexCAN block, with no power consumption. It is requested by the CPU through the assertion of MCR[MDIS], and the acknowledgement is obtained through the assertion by the FlexCAN of MCR[LPMACK]. The CPU must only consider the FlexCAN in Disable mode when both request and acknowledgement conditions are satisfied.

If the module is disabled during Freeze mode, it requests to disable the clocks to the PE and CHI submodules, sets MCR[LPMACK] and negates MCR[FRZACK].

If the module is disabled during transmission or reception, FlexCAN does the following:

- Waits to be in either Idle or Bus Off state, or else waits for the third bit of Intermission and then checks it to be recessive.
- Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. A pending move-in is not taken into account.
- Ignores its Rx input pin and drives its Tx pin as recessive.
- Shuts down the clocks to the PE and CHI submodules
- Sets MCR[NOTRDY] and MCR[LPMACK].

The Bus Interface Unit continues to operate, enabling the CPU to access memory-mapped registers, except the Rx Mailboxes Global Mask registers, the Rx Buffer 14 Mask register, the Rx Buffer 15 Mask register, the Legacy Rx FIFO Global Mask register. The Legacy Rx FIFO Information register, the message buffers, the Rx Individual Mask registers, and the reserved words within RAM may not be accessed when the module is in Disable mode. Exiting from this mode is done by negating the MDIS bit by the CPU, which causes the FlexCAN to request to resume the clocks and negate the LPMACK bit after the CAN protocol engine recognizes the negation of disable mode requested by the CPU.

70.3.2 Transmit process

To transmit a CAN frame, the CPU must prepare a message buffer for transmission by executing the following procedure:

1. Check whether the respective interrupt bit is set and clear it.
2. If the MB is active (transmission pending), write the ABORT code (1001b) to the CODE field of the Control and Status word to request an abort of the transmission.
3. Wait for the corresponding IFLAG bit to be asserted by polling the IFLAG register, or by the interrupt request if enabled by the respective IMASK bit.
4. Read back the CODE field to check if the transmission was aborted or transmitted (see Transmission abort mechanism).
5. Clear the corresponding interrupt flag.
6. Write the ID register (containing the local priority if enabled via MCR[LPRIO_EN]).
7. Write payload data bytes.

8. Configure the Control and Status word with the desired configuration.
    a. Set ID type via MB_CS[IDE].
    b. Set Remote Transmission Request (if needed) via MB_CS[RTR].
    c. If MCR[FDEN] is enabled, also configure the MB_CS[EDL] and MB_CS[BRS] fields. For details about the relationship between the written value and transmitted value of the MB_CS[ESI] field, see Table 425.[7]
    d. Set Data Length Code in bytes via MB_CS[DLC]. See Table 449 for detailed information.
    e. Activate the message buffer to transmit the CAN frame by setting MB_CS[CODE] to Ch.

NOTE

It is strongly recommended that all the fields in MB_CS word be configured in only one 32-bit write operation to maximize software performance. If the fields are configured in separate writes, the MB_CS[CODE] must be the last write in the C/S word.

When the MB is activated, it participates in the arbitration process and is eventually transmitted according to its priority. When the DLC value stored in the MB selected for transmission is larger than the respective MB payload size, FlexCAN adds the necessary number of bytes with constant CCh pattern to complete the expected DLC.

At the end of the successful transmission:

- The value of the free running timer is written into the Time Stamp field.
- The CODE field in the Control and Status word is updated.
- Both CRC and FDCRC registers are updated.
- A status flag is set in the Interrupt Flag register.
- An interrupt is generated if allowed by the corresponding Interrupt Mask Register bit.

The new CODE field after transmission depends on the code that was used to activate the MB (see Table 446 and Table 447 in Message buffer structure).

When the Abort feature is enabled (MCR[AEN] is asserted), after the Interrupt flag is asserted for a mailbox configured as transmit buffer, the mailbox is blocked. Therefore the CPU is not able to update it until the Interrupt Flag is negated by the CPU. This means that the CPU must clear the corresponding IFLAG bit before starting to prepare this MB for a new transmission or reception.

NOTE

If backwards compatibility is desired (MCR[AEN] is negated), write the INACTIVE code (1000b) to the CODE field to inactivate the MB. However, in this case the pending frame may be transmitted without notification (see Mailbox inactivation).

70.3.3 Arbitration process

The arbitration process scans the mailboxes, searching for the Tx mailbox that holds the message to be sent in the next opportunity. This mailbox is called the arbitration winner.

The scan starts from the lowest number mailbox and runs toward the higher ones.

The arbitration process is triggered in the following events:

- From the CRC field of the CAN frame. The start point depends on the value of CTRL2[TASD].
- During the Error Delimiter field of a CAN frame.
- During the Overload Delimiter field of a CAN frame.
- When the winner is inactivated and the CAN bus has still not reached the first bit of the Intermission field.

[7] The ESI field does not need to be written, and will automatically be transmitted dominant by error active nodes and recessive by error passive nodes. Note that there is an exception if the FlexCAN is operating as a network gateway: In that case, the CPU writes the MB_CS[ESI] bit according to the error status of the node which sent the message.

- When there is a CPU write to the C/S word of a winner MB and the CAN bus has still not reached the first bit of the Intermission field.
- When CHI is in Idle state and the CPU writes to the C/S word of any MB.
- When FlexCAN exits Bus Off state.
- Upon leaving Freeze mode or Low Power mode.

If the arbitration process does not manage to evaluate all mailboxes before the CAN bus has reached the first bit of the Intermission field, the temporary arbitration winner is invalidated and the FlexCAN will not compete for the CAN bus in the next opportunity.

The arbitration process selects the winner among the active Tx mailboxes at the end of the scan according to the settings of both CTRL1[LBUF] and MCR[LPRIOEN].

70.3.3.1 Lowest-number mailbox first

If CTRL1[LBUF] is asserted, the first (lowest number) active Tx mailbox found is the arbitration winner. MCR[LPRIOEN] has no effect when CTRL1[LBUF] is asserted.

70.3.3.2 Highest-priority mailbox first

If CTRL1[LBUF] is negated, then the arbitration process searches the active Tx mailbox with the highest priority, which means that this mailbox's frame would have a higher probability to win the arbitration on CAN bus when multiple external nodes compete for the bus at the same time.

The sequence of bits considered for this arbitration is called the *arbitration value* of the mailbox. The highest-priority Tx mailbox is the one that has the lowest arbitration value among all Tx mailboxes.

If two or more mailboxes have equivalent arbitration values, the mailbox with the lowest number is the arbitration winner.

The composition of the arbitration value depends on the MCR[LPRIOEN] setting.

70.3.3.2.1 Local priority disabled

If MCR[LPRIOEN] is negated the arbitration value is built in the exact sequence of bits as they would be transmitted in a CAN frame (see the following table) in such a way that the local priority is disabled.

Table 415. Composition of the arbitration value when local priority is disabled

Format	Mailbox arbitration value (32 bits)				
Standard (IDE = 0)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	- (18 bits)	- (1 bit)
Extended (IDE = 1)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

70.3.3.2.2 Local priority enabled

To enable local priority, MCR[LPRIOEN] must be asserted. In this case the mailbox PRIO field is included at the very left of the arbitration value (see the following table).

Table 416. Composition of the arbitration value when local priority is enabled

Format	Mailbox arbitration value (35 bits)					
Standard (IDE = 0)	PRIO (3 bits)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	- (18 bits)	- (1 bit)
Extended (IDE = 1)	PRIO (3 bits)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

Because the PRIO field is the most significant part of the arbitration value, mailboxes with low PRIO values have higher priority than mailboxes with high PRIO values regardless of the rest of their arbitration values.

Note that the PRIO field is not part of the frame on the CAN bus. Its purpose is only to affect the internal arbitration process.

70.3.3.3 Arbitration process (continued)

After the arbitration winner is found, its content is copied to a hidden auxiliary MB called Tx Serial Message Buffer (Tx SMB), which has the same structure as a normal MB but is not user accessible. This operation is called move-out and after it is done, write access to the C/S word of the corresponding MB is blocked (if MCR[AEN] is asserted). Write access is restored in the following events:

- After the MB is transmitted and the corresponding IFLAG bit is cleared by the CPU.
- FlexCAN enters Freeze mode or Bus Off.
- FlexCAN loses the bus arbitration or there is an error during the transmission.

At the first opportunity window on the CAN bus, the message on the Tx SMB is transmitted according to the CAN protocol rules.

Arbitration process can be triggered in the following situations:

- During Rx and Tx frames from CAN CRC field to end of frame. CTRL2[TASD] value may be changed to optimize the arbitration start point.
- During CAN Bus Off state from TX_ERR_CNT=124 to 128. CTRL2[TASD] value may be changed to optimize the arbitration start point.
- During C/S write by CPU in Bus Idle. First C/S write starts arbitration process, and a second C/S write during this same arbitration restarts the process. If other C/S writes are performed, Tx arbitration process is pending. If there is no arbitration winner after the arbitration process has finished, then the TX arbitration machine begins a new arbitration process. If there is a pending arbitration and Bus Idle state starts, then an arbitration process is triggered. In this case the first and second C/S write in Bus Idle will not restart the arbitration process. It is possible that there is not enough time to finish arbitration in Wait For Bus Idle state and the next state is Idle. In this case the scan is not interrupted, and it is completed during Bus Idle state. During this arbitration C/S write does not cause arbitration restart.
- Arbitration winner deactivation during a valid arbitration window.
- Upon exiting Freeze mode (first bit of the Wait For Bus Idle state). If there is a re-synchronization during Wait For Bus Idle, the arbitration process is restarted.

Arbitration process stops in the following situations:

- All mailboxes were scanned.
- A Tx active mailbox is found if lowest buffer feature is enabled.
- Arbitration winner inactivation or abort during any arbitration process.
- There was not enough time to finish Tx arbitration process (for instance, when a deactivation was performed near the end of frame). In this case arbitration process is pending.
- Error or overload flag in the bus.
- Low Power or Freeze mode request in Idle state.

Arbitration is considered pending as described below:

- It was not possible to finish arbitration process in time.
- C/S write during arbitration if write is performed in a MB whose number is lower than the Tx arbitration pointer.
- Any C/S write if there is no Tx arbitration process in progress.
- Rx Match has just updated a Rx code to Tx code.
- Entering Bus Off state.

C/S write during arbitration has the following effect:

- If C/S write is performed in the arbitration winner, a new process is restarted immediately.
- If C/S write is performed in a MB whose number is higher than the Tx arbitration pointer, the ongoing arbitration process will scan this MB as normal.

70.3.4 Receive process

To be able to receive CAN frames into a mailbox, the CPU must prepare it for reception by executing the following steps:

1. If the mailbox is active (either Tx or Rx) inactivate the mailbox (see Mailbox inactivation), preferably with a safe inactivation (see Transmission abort mechanism).
2. Write the ID word into the mailbox.
3. Write the EMPTY code (0100b) to the CODE field of the Control and Status word to activate the mailbox. No setup is required for EDL, BRS, and ESI bits—they are overwritten by the respective bit fields in the received message.

After the MB is activated, it will be able to receive frames that match the programmed filter. At the end of a successful reception, the mailbox is updated by the move-in process (see Move-in) as follows:

1. The received data field (8 bytes at most for Classical CAN message format and up to 64 bytes for CAN FD message format) is stored.
2. The received Identifier field is stored.
3. The value of the Free Running Timer at the time of the second bit of frame's Identifier field is written into the mailbox's Time Stamp field.
4. The received SRR, IDE, RTR, EDL, BRS, ESI, and DLC fields are stored.
5. The CODE field in the Control and Status word is updated (see Table 446 and Table 447 in Section Message buffer structure).
6. A status flag is set in the Interrupt Flag Register and an interrupt is generated if allowed by the corresponding Interrupt Mask Register bit.

The recommended way for the CPU to service (read) the frame received in a mailbox is by the following procedure:

1. Read the Control and Status word of that mailbox.
2. Check if the BUSY bit is deasserted, indicating that the mailbox is locked. Repeat step 1) while it is asserted. See Mailbox lock mechanism.
3. Read the contents of the mailbox. After the mailbox is locked, its contents won't be modified by FlexCAN move-in processes. See Move-in.
4. Acknowledge the proper flag at IFLAG registers.
5. Read the free running timer to unlock the mailbox.

The CPU should poll for frame reception by the status flag bit for the specific mailbox in one of the IFLAG registers and not by the CODE field of that mailbox. Polling the CODE field does not work because after a frame is received and the CPU services the mailbox (by reading the C/S word followed by unlocking the mailbox), the CODE field will not return to EMPTY. It will remain FULL, as explained in Table 446. If the CPU tries to work around this behavior by writing to the C/S word to force an EMPTY code after reading the mailbox without a prior safe inactivation, a newly received frame matching the filter of that mailbox may be lost.

CAUTION

In summary: never do polling by reading directly the C/S word of the mailboxes. Instead, read the IFLAG registers.

Note that the received frame's Identifier field is always stored in the matching mailbox, thus the contents of the ID field in a mailbox may change if the match was due to masking. When MCR[SRXDIS] is asserted, FlexCAN will not store frames transmitted by itself in any MB, even if it contains a matching Rx mailbox, and no interrupt flag or interrupt signal will be generated. Otherwise, when MCR[SRXDIS] is deasserted, FlexCAN can receive frames transmitted by itself if a matching Rx mailbox exists.

To be able to receive CAN frames through the Legacy Rx FIFO, the CPU must enable and configure the Legacy Rx FIFO during Freeze mode (see Legacy Rx FIFO). Upon receiving the Frames Available in Legacy Rx FIFO interrupt (see the description of IFLAG1[BUF5I] "Frames available in Legacy Rx FIFO"), the CPU should service the received frame using the following procedure:

1. Read the Control and Status word (optional: needed only if a mask was used for IDE and RTR bits).
2. Read the ID field (optional: needed only if a mask was used).
3. Read the data field.
4. Read the RXFIR register. (This step is optional, and is executed only if the IDHIT (Identifier Acceptance Filter Hit Indicator) is required in the application.)
5. Clear the Frames Available in Legacy Rx FIFO interrupt by writing one to IFLAG1[BUF5I] (mandatory: releases the MB and allows the CPU to read the next Rx FIFO entry).

When MCR[DMA] is asserted, upon receiving a frame in Legacy FIFO, IFLAG1[BUF5I] generates a DMA request and does not generate a CPU interrupt (see Legacy Rx FIFO under DMA operation). The IMASK1 bits in Legacy Rx FIFO region are not used.

The DMA controller must service the received frame using the following procedure:

1. Read the Control and Status word (read 80h address, optional).
2. Read the ID field (read 84h address, optional).
3. Read all data bytes (start read at 88h address, optional).
4. Read the last data bytes (read 8Ch address is mandatory).

70.3.5 Matching process

The matching process scans the MB memory looking for Rx MBs programmed with the same ID as the one received from the CAN bus. If the Legacy Rx or Enhanced Rx FIFO is enabled, the priority of scanning can be selected between mailboxes and FIFO filters. The matching starts from the lowest number message buffer toward the higher ones. If no match is found within the first structure then the other is scanned subsequently. In the event that the FIFO is full, the matching algorithm always looks for a matching MB outside the FIFO region.

For enhanced Rx FIFO see Enhanced Rx FIFO.

For legacy Rx FIFO see Legacy Rx FIFO.

As the frame is being received, it is stored in a hidden auxiliary MB called Rx Serial Message Buffer (Rx SMB).

The matching process start point depends on the following conditions:

- If the received frame is a remote frame, the start point is the CRC field of the frame.
- If the received frame is a data frame with DLC field equal to zero, the start point is the CRC field of the frame.
- If the received frame is a data frame with DLC field different than zero, the start point is the DATA field of the frame.

If a matching ID is found in the FIFO table or in one of the mailboxes, the contents of the Rx SMB are transferred to the FIFO or to the matched mailbox by the move-in process. If any CAN protocol error is detected then no match results are transferred to the FIFO or to the matched mailbox at the end of reception.

The matching process scans all matching elements of both Rx FIFO (if enabled) and the active Rx mailboxes (CODE is EMPTY, FULL, OVERRUN, or RANSWER) in search of a successful comparison with the matching elements of the Rx SMB that is receiving the frame on the CAN bus. The Rx SMB has the same structure as a mailbox. The reception structures (Rx FIFO or mailboxes) associated with the matching elements that had a successful comparison are the *matched structures*. The *matching winner* is selected at the end of the scan among those matched structures and depends on conditions described ahead. See the following table.

Table 417. Matching architecture

Structure	SMB[RTR]	CTRL2[RRS]	CTRL2[EACEN]	MB[IDE]	MB[RTR]	MB[ID[1]]	MB[CODE]
Mailbox	0	—	0	cmp[2]	no_cmp[3]	cmp_msk[4]	EMPTY or FULL or OVERRUN
Mailbox	0	—	1	cmp_msk	cmp_msk	cmp_msk	EMPTY or FULL or OVERRUN
Mailbox	1	0	—	cmp	no_cmp	cmp	RANSWER
Mailbox	1	1	0	cmp	no_cmp	cmp_msk	EMPTY or FULL or OVERRUN
Mailbox	1	1	1	cmp_msk	cmp_msk	cmp_msk	EMPTY or FULL or OVERRUN
Legacy Rx FIFO[5]	—	—	—	cmp_msk	cmp_msk	cmp_msk	—

1. For mailbox structure, If SMB[IDE] is asserted, the ID is 29 bits (ID Standard + ID Extended). If SMB[IDE] is negated, the ID is only 11 bits (ID Standard). For Legacy Rx FIFO structure, the ID depends on IDAM.
2. cmp: Compares the Rx SMB contents with the MB contents regardless the masks.
3. no_cmp: The Rx SMB contents are not compared with the MB contents.
4. cmp_msk: Compares the Rx SMB contents with MB contents taking into account the masks.
5. SMB[IDE] and SMB[RTR] are not taken into account when IDAM is type C.

NOTE

For Enhanced Rx FIFO, see Enhanced Rx FIFO matching process.

A reception structure is free-to-receive when any of the following conditions is satisfied:

- The CODE field of the mailbox is EMPTY.
- The CODE field of the mailbox is either FULL or OVERRUN and it has already been serviced (the C/S word was read by the CPU and unlocked as described in Mailbox lock mechanism).
- The CODE field of the mailbox is either FULL or OVERRUN and an inactivation (see Mailbox inactivation) is performed.
- The Legacy Rx FIFO or Enhanced Rx FIFO is not full.

The scan order for mailboxes and Legacy Rx FIFO is from the matching element with lowest number to the higher ones.

The matching winner search for mailboxes is affected by MCR[IRMQ]. If it is negated, the matching winner is the first matched mailbox regardless if it is free-to-receive or not. If it is asserted, the matching winner is selected according to the priority below:

1. the first free-to-receive matched mailbox;
2. the last non free-to-receive matched mailbox.

It is possible to select the priority of scan between mailboxes and Legacy Rx FIFO or Enhanced Rx FIFO with CTRL2[MRP].

If the selected priority is Rx FIFO first:

- If the Rx FIFO is a matched structure and is free-to-receive, then the Rx FIFO is the matching winner regardless of the scan for mailboxes.
- Otherwise (the Rx FIFO is not a matched structure or is not free-to-receive), then the matching winner is searched among mailboxes as described above.

If the selected priority is mailboxes first:

- If a free-to-receive matched mailbox is found, it is the matching winner regardless of the scan for Rx FIFO.
- If no matched mailbox is found, then the matching winner is searched in the scan for the Rx FIFO.
- If both conditions above are not satisfied and a non–free-to-receive matched mailbox is found, then the matching winner determination is conditioned by MCR[IRMQ]:
  - If MCR[IRMQ] is negated, the matching winner is the first matched mailbox.
  - If MCR[IRMQ] is asserted, the matching winner is the Rx FIFO if it is a free-to-receive matched structure; otherwise, the matching winner is the last non–free-to-receive matched mailbox.

See the following table for a summary of matching possibilities.

Table 418. Matching possibilities and resulting reception structures

RFEN or ERFEN	IRMQ	MRP	Matched in MB	Matched in FIFO	Reception structure	Description
No FIFO, only MB, match is always MB first						
0	0	X[1]	None[2]	—[3]	None	Frame lost by no match
0	0	X	Free[4]	—	First MB	
0	1	X	None	—	None	Frame lost by no match
0	1	X	Free	—	First MB	
0	1	X	Not free	—	Last MB	Overrun
FIFO enabled, no match in FIFO is as if FIFO does not exist						
1	0	X	None	None[5]	None	Frame lost by no match
1	0	X	Free	None	First MB	
1	1	X	None	None	None	Frame lost by no match
1	1	X	Free	None	First MB	
1	1	X	Not free	None	Last MB	Overrun
FIFO enabled, Queue disabled						
1	0	0	X	Not full[6]	FIFO	
1	0	0	None	Full[7]	None	Frame lost by FIFO full (FIFO overflow)
1	0	0	Free	Full	First MB	
1	0	0	Not free	Full	First MB	
1	0	1	None	Not full	FIFO	
1	0	1	None	Full	None	Frame lost by FIFO full (FIFO overflow)

Table continues on the next page...

Table 418. Matching possibilities and resulting reception structures (continued)

RFEN or ERFEN	IRMQ	MRP	Matched in MB	Matched in FIFO	Reception structure	Description
1	0	1	Free	X	First MB	
1	0	1	Not free	X	First MB	Overrun
FIFO enabled, queue enabled						
1	1	0	X	Not full	FIFO	
1	1	0	None	Full	None	Frame lost by FIFO full (FIFO overflow)
1	1	0	Free	Full	First MB	
1	1	0	Not free	Full	Last MB	Overrun
1	1	1	None	Not full	FIFO	
1	1	1	Free	X	First MB	
1	1	1	Not free	Not full	FIFO	
1	1	1	Not free	Full	Last MB	Overrun

1. This is a don't care condition.
2. Matched in MB "None" means that the frame has not matched any MB (free-to-receive or non–free-to-receive).
3. This is a forbidden condition.
4. Matched in MB "Free" means that the frame matched at least one MB free-to-receive regardless of whether it has matched MBs non-free-to-receive.
5. Matched in FIFO "None" means that the frame has not matched any filter in FIFO. It is as if the FIFO didn't exist (CTRL2[RFEN]=0 and ERFCR[ERFEN]=0).
6. Matched in FIFO "Not full" means that the frame has matched a FIFO filter and has empty slots to receive it.
7. Matched in FIFO "Full" means that the frame has matched a FIFO filter but couldn't store it because it has no empty slots to receive it.

If a non-safe mailbox inactivation (see Mailbox inactivation) occurs during matching process and the mailbox inactivated is the temporary matching winner, then the temporary matching winner is invalidated. The matching elements scan is not stopped nor restarted—it continues normally. The consequence is that the current matching process works as if the matching elements compared before the inactivation did not exist, therefore a message may be lost.

Suppose, for example, that the FIFO is disabled, IRMQ is enabled, there are two MBs with the same ID, and FlexCAN starts receiving messages with that ID. Let us say that these MBs are the second and the fifth in the array. When the first message arrives, the matching algorithm finds the first match in MB number 2. The code of this MB is EMPTY, so the message is stored there. When the second message arrives, the matching algorithm finds MB number 2 again, but it is not "free-to-receive", so it keeps looking, finds MB number 5 and stores the message there. If yet another message with the same ID arrives, the matching algorithm finds out that there are no matching MBs that are "free-to-receive", so it decides to overwrite the last matched MB, which is number 5. In doing so, it sets the CODE field of the MB to indicate OVERRUN.

The ability to match the same ID in more than one MB can be exploited to implement a reception queue (in addition to the full featured FIFO) to allow more time for the CPU to service the MBs. By programming more than one MB with the same ID, received messages are queued into the MBs. The CPU can examine the Time Stamp field of the MBs to determine the order in which the messages arrived.

Matching to a range of IDs is possible by using ID acceptance masks. FlexCAN supports individual masking per MB (see the description of Rx Individual Mask Registers (RXIMR0 - RXIMR95)). During the matching algorithm, if a mask bit is asserted, then the corresponding ID bit is compared. If the mask bit is negated, the corresponding ID bit is a "don't care". Note that the Individual Mask Registers are implemented in RAM, so they are not initialized out of reset. Also, they can only be programmed when the module is in Freeze mode; otherwise, they are blocked by hardware.

FlexCAN also supports an alternate masking scheme with only four mask registers (RXFGMASK, RXMGMASK, RX14MASK, and RX15MASK) for backwards compatibility with legacy applications. This alternate masking scheme is enabled when the IRMQ bit in the MCR register is negated.

70.3.6 Move process

There are two types of move process: move-in and move-out.

70.3.6.1 Move-in

The move-in process is the copy of a message received by an Rx SMB to an Rx mailbox or FIFO that has matched it. If the move destination is the Legacy Rx FIFO, attributes of the message are also copied to the CAN_RXFIR FIFO. Each Rx SMB has its own move-in process, but only one is performed at a given time as described ahead. The move-in starts only when the message held by the Rx SMB has a corresponding matching winner (see Matching process) and all of the following conditions are true:

- The CAN bus has either reached or already gone past:
  - The second bit of Intermission field next to the frame that carried the message that is in the Rx SMB.
  - The first bit of an overload frame next to the frame that carried the message that is in the Rx SMB.
- There is no ongoing matching process.
- The destination mailbox is not locked by the CPU.
- There is no ongoing move-in process from another Rx SMB. If more than one move-in process is to be started at the same time, both are performed and the newest substitutes for the oldest.

The term pending move-in is used throughout the documentation and stands for a move-to-be that still does not satisfy all of the aforementioned conditions.

The move-in is cancelled and the Rx SMB is able to receive another message if any of the following conditions is satisfied:

- The destination mailbox is inactivated after the CAN bus has reached the first bit of Intermission field next to the frame that carried the message and its matching process has finished.
- There is a previous pending move-in to the same destination mailbox.
- The Rx SMB is receiving a frame transmitted by the FlexCAN itself and self-reception is disabled (MCR[SRXDIS] is asserted).
- Any CAN protocol error is detected.

Note that the pending move-in is not cancelled if the module enters Freeze or Low-Power mode. It stays on hold, only waiting for Freeze and Low-Power mode to be exited and the module to be unlocked. If an MB is unlocked during Freeze mode, the move-in happens immediately.

The move-in process is the execution by the FlexCAN of the following steps:

1. Push IDHIT into the RXFIR FIFO if the message is destined for the Legacy Rx FIFO.
2. Read all data words from the Rx SMB in accordance with the selected payload size for the Rx storage element.
3. Write all data words to the Rx mailbox in accordance with the selected payload size for the Rx storage element. If the data size of the storage element is smaller than the original payload size described in the message's DLC field, the payload is truncated and the high order bytes that do not fit the destination size are lost.
4. Read the Control/Status and ID words from the Rx SMB.
5. Write Control/Status and ID words to the Rx mailbox, and update the CODE field.

The move-in process is not atomic, in such a way that it is immediately cancelled by the inactivation of the destination mailbox (see Mailbox inactivation). In this case the mailbox may remain partially updated, thus incoherent. The exception is if the move-in destination is a Legacy or Enhanced Rx FIFO message buffer; then the process cannot be cancelled.

The BUSY Bit (least significant bit of the CODE field) of the destination message buffer is asserted while the move-in is being performed to alert the CPU that the message buffer content is temporarily incoherent.

70.3.6.2 Move-out

The move-out process is the copy of the content from a Tx Mailbox to the Tx SMB when a message for transmission is available (see Arbitration process). The move-out occurs in the following conditions:

- The first bit of Intermission field
- During Bus Off state when TX Error Counter is in the 124 to 128 range
- During Bus Idle state
- During Wait For Bus Idle state

The move-out process is not atomic. Only the CPU has priority to access the memory concurrently out of Bus Idle state. In Bus Idle, the move-out has the lowest priority to the concurrent memory accesses.

70.3.7 Data coherence

In order to maintain data coherency and FlexCAN proper operation, the CPU must obey the rules described in Transmit process and Receive process.

70.3.7.1 Transmission abort mechanism

The abort mechanism provides a safe way to request the abort of a pending transmission. A feedback mechanism is provided to inform the CPU if the transmission was aborted or if the frame could not be aborted and was transmitted instead.

Two primary conditions must be fulfilled in order to abort a transmission:

- MCR[AEN] must be asserted.
- The first CPU action must be the writing of abort code (1001b) into the CODE field of the Control and Status word.

Active MBs configured for transmission must be aborted first before they can be updated. If the abort code is written to a Mailbox that is currently being transmitted or to a Mailbox that was already loaded into the Tx SMB for transmission, the write operation is blocked and the transmission is not disturbed. However, the abort request is captured and kept pending until one of the following conditions is satisfied:

- The module loses the bus arbitration.
- There is an error during the transmission.
- The module is put into Freeze mode.
- The module enters Bus Off state.
- There is an overload frame.

If none of the conditions above are reached:

- The MB is transmitted correctly.
- The interrupt flag is set in the IFLAG register.
- An interrupt to the CPU is generated (if enabled).

The abort request is automatically cleared when the interrupt flag is set. On the other hand, if only one of the above conditions is reached, the frame is not transmitted; therefore:

- The abort code is written into the CODE field.
- The interrupt flag is set in the IFLAG.
- An interrupt is (optionally) generated to the CPU.

If the CPU writes the abort code before the transmission begins internally, then the write operation is not blocked; therefore, the MB is updated and the interrupt flag is set. In this way the CPU only needs to read the abort code to make sure the active MB was safely inactivated. Although the AEN bit is asserted and the CPU wrote the abort code, in this case the MB is inactivated and not aborted, because the transmission did not start yet. One Mailbox is aborted only when the abort request is captured and kept pending until one of the previous conditions is satisfied.

70.3.7.2 Mailbox inactivation

Inactivation is a mechanism provided to protect the mailbox against updates by the FlexCAN internal processes, thus allowing the CPU to rely on mailbox data coherence after having updated it, even in Normal mode.

Inactivation of transmission mailboxes must be performed just when MCR[AEN] is deasserted.

If a mailbox is inactivated, it participates in neither the arbitration process nor the matching process until it is reactivated. See Transmit process and Receive process for more detailed instructions on how to inactivate and reactivate a mailbox.

To inactivate a mailbox, the CPU must update its CODE field to INACTIVE (either 0b or 1000b).

Because you will not be able to synchronize the CODE field update with the FlexCAN internal processes, an inactivation can have the following consequences:

- A frame in the bus that matches the filtering of the inactivated Rx mailbox may be lost without notice, even if there are other mailboxes with the same filter.
- A frame containing the message within the inactivated Tx mailbox may be transmitted without setting the respective IFLAG.

In order to perform a safe inactivation and avoid the above consequences for Tx mailboxes, the CPU must use the transmission abort mechanism (see Transmission abort mechanism).

The inactivation automatically unlocks the mailbox (see Mailbox lock mechanism).

NOTE

Message buffers that are part of the Legacy Rx FIFO or Enhanced Rx FIFO cannot be inactivated. There is no write protection on the Legacy FIFO region by FlexCAN. CPU must maintain data coherency in the Legacy FIFO region when RFEN is asserted.

70.3.7.3 Mailbox lock mechanism

Other than mailbox inactivation, FlexCAN has another data coherence mechanism for the receive process. When the CPU reads the Control and Status word of an Rx MB with codes FULL or OVERRUN, FlexCAN assumes that the CPU wants to read the whole MB in an atomic operation, and therefore it sets an internal lock flag for that MB. The lock is released when the CPU reads the free running timer (global unlock operation), or when it reads the Control and Status word of another MB regardless of its code. A CPU write into the C/S word also unlocks the MB, but this procedure is not recommended for normal unlock use because it cancels a pending move and potentially may lose a received message. The MB locking prevents a new frame from being written into the MB while the CPU is reading it.

NOTE

The locking mechanism applies only to Rx MBs that are not part of the Legacy Rx FIFO and have a code different than INACTIVE (0b) or EMPTY[1] (0100b). Also, Tx MBs cannot be locked.

Suppose, for example, that the Legacy Rx FIFO is disabled and the second and the fifth MBs of the array are programmed with the same ID, and FlexCAN has already received and stored messages into these two MBs. Suppose now that the CPU decides to read MB number 5 and at the same time another message with the same ID is arriving. When the CPU reads the Control and Status word of MB number 5, this MB is locked. The new message arrives and the matching algorithm finds out that there are no free-to-receive MBs, so it decides to override MB number 5. However, this MB is locked, so the new message cannot be written there. It will remain in the Rx SMB waiting for the MB to be unlocked, and only then will be written to the MB.

If the MB is not unlocked in time and yet another new message with the same ID arrives, then the new message overwrites the one on the Rx SMB and there will be no indication of lost messages either in the CODE field of the MB or in the Error and Status Register.

While the message is being moved in from the Rx SMB to the MB, the BUSY bit on the CODE field is asserted. If the CPU reads the Control and Status word and finds out that the BUSY bit is set, it should defer accessing the MB until the BUSY bit is negated.

[1] In previous FlexCAN versions, reading the C/S word locked the MB even if it was EMPTY. This behavior is maintained when the IRMQ bit is negated.

NOTE

If the BUSY bit is asserted or if the MB is empty, then reading the Control and Status word does not lock the MB.

Inactivation takes precedence over locking. If the CPU inactivates a locked Rx MB, then its lock status is negated and the MB is marked as invalid for the current matching round. Any pending message on the Rx SMB will not be transferred anymore to the MB. An MB is unlocked when the CPU reads the Free Running Timer Register (see Free Running Timer (TIMER)), or the C/S word of another MB.

Lock and unlock mechanisms have the same functionality in both Normal and Freeze modes.

An unlock during Normal or Freeze mode results in the move-in of the pending message. However, the move-in is postponed if an unlock occurs during a low power mode (see Modes of operation), and it takes place only when the module returns to Normal or Freeze modes.

70.3.8 Enhanced Rx FIFO

FlexCAN supports an enhanced Rx FIFO engine which can store up to 20 CAN FD messages. The region 2000h–204Fh contains the output of the FIFO which should be read by the CPU. The enhanced Rx FIFO is enabled by setting EFRCR[ERFEN]. Although FlexCAN has two FIFO options, Legacy Rx FIFO and Enhanced Rx FIFO, they cannot be enabled at the same time. Refer to Legacy Rx FIFO to find detailed information about Legacy Rx FIFO.

The enhanced Rx FIFO has a watermark which is configured by setting EFRCR[ERFWM]. If EFRCR[ERFWM] is set, the CPU is notified only if a minimum number of messages is stored in the FIFO. When the number of stored messages is greater than the value in EFRCR[ERFWM], ERFSR[ERFWMI] is set by the hardware. Optionally, an interrupt or a DMA transfer can be triggered if MCR[DMA] or ERFIER[ERFWMIIE] are set, respectively.

For the Enhanced Rx FIFO to receive, the CPU must execute the configuration procedure below. The same procedure must be done if the CPU needs to change any of the configuration of the Enhanced RX FIFO.

1. Enter Freeze mode.
2. Enable Enhanced Rx FIFO by setting ERFCR[ERFEN], if it is not already enabled.[8]
3. Write one to ERFSR[ERFCLR] to reset Enhanced Rx FIFO engine.
4. Clear EFRSR[ERFUFW], EFRSR[ERFOVF], EFRSR[ERFWMI], and EFRSR[ERFDA], if they are set.
5. Write EFRCR[NFE] to configure the total number of enhanced Rx FIFO filter elements to be used in Enhanced Rx FIFO reception.
6. Write ERFCR[NEXIF] to configure the number of extended ID and standard ID filter elements to be used in Enhanced Rx FIFO reception.[9]
7. Configure the Enhanced Rx FIFO watermark by writing ERFCR[ERFWM]. If MCR[DMA] is set, ERFCR[ERFWM] should be configures as 0x0.
8. If interrupts will be used, set the interrupt enables in the ERIER register.
9. If DMA will be used, set MCR[DMA] to enable DMA operation and write ERFCR[DMALW] to configure the number of words to transfer for each Enhanced Rx FIFO data element.
10. Configure the filter elements by writing in the ERFFEL*n* registers.[10]
11. Leave Freeze mode.

There are two types of Enhanced Rx FIFO filter elements that can be stored in ERFFEL*n* registers: extended-ID filter element and standard-ID filter element. Each extended-ID filter element is stored in two ERFFEL*n* registers while each standard-ID filter element is stored in one ERFFEL*n* register. The total number of Enhanced RX FIFO filter elements is defined by ERFCR[NFE].

In addition, the filter memory space can be split into two regions: one for extended-ID filter elements and another for standard-ID filter elements, according to ERFCR[NEXIF]. The figure below shows how the Enhanced Rx filter elements are defined. Detailed

[8] MCR[RFEN] must be cleared so that Enhanced Rx FIFO is enabled.
[9] NEXIF ≤ NFE + 1.
[10] ERFFEL*n* registers are implemented in RAM; thus they must be explicitly initialized prior to any reception.

information about the Enhanced Rx FIFO matching process and filter element formats can be found in Enhanced Rx FIFO matching process.

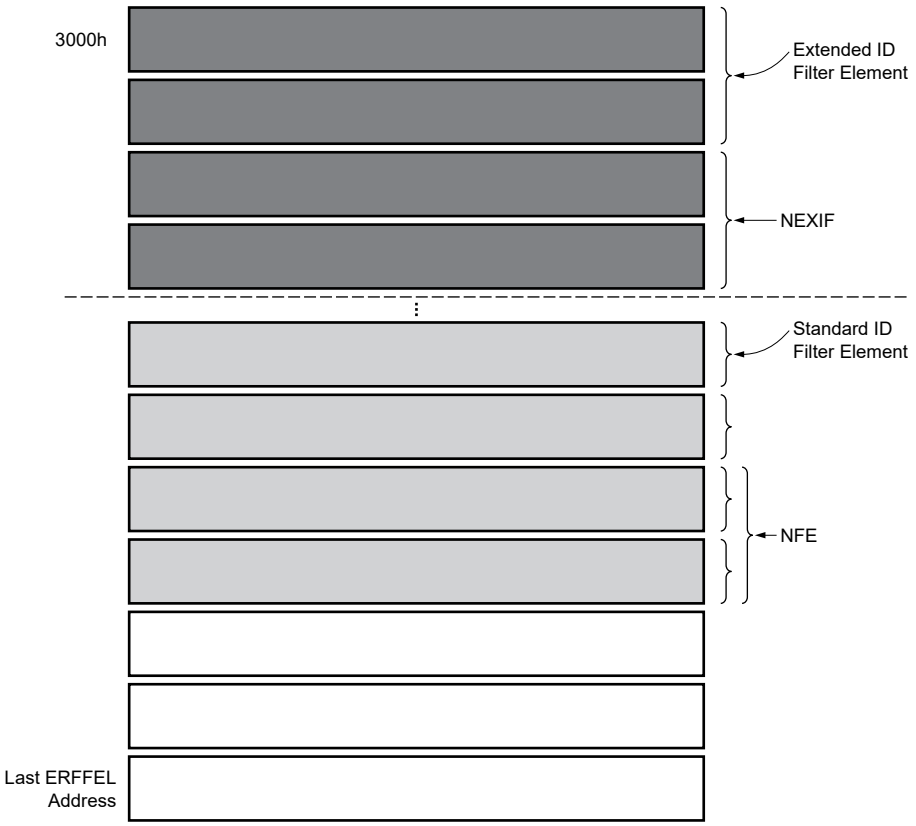


Figure 373. Number of enhanced Rx FIFO filter elemets

70.3.8.1 Enhanced Rx FIFO matching process

When ERFCR[ERFEN] is enabled, FlexCAN scans the ERFFEL*n* memory region. If at least one filter element satisfies the matching criteria, then the CAN message content is transfered to the enhanced RX FIFO memory.

NOTE

If multiple filters match the incoming message ID, then the first matching filter found by the matching process must be indicated in IDHIT.

Each ERFFEL*n* register can store one standard filter element. The matching criteria are defined by ERFFEL*n*[FSCH] in this way:

- If FSCH = b00, the filter scheme is based on mask and filter. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base frame format (IDE = 0)
  2. (ID[*n*] = STD ID filter [*n*]) or (STD ID Mask[*n*] =0) for each bit *n* from 0 to 10
  3. (RTR = RTR Filter) or (RTR MASK = 0)

In this explanation, RTR and ID are the Remote Transmit Request field and the ID from a CAN message, respectively.

If FSCH = b00, the filters and masks are defined as shown in this table.

Table 419. Standard ID filter element with filter + mask scheme (FSCH = b00)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH=b00		Reserved		RTR filter	STD ID filter		Reserved		RTR mask	STD ID mask	

- If FSCH = b01, the filter scheme is based on range. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base frame format (IDE = 0)
  2. ID ≥ STD ID Filter1
  3. ID ≤ STD ID Filter2
  4. (RTR = RTR filter) or (RTR MASK = 0)

RTR and ID are the Remote Transmit Request bit and ID from a CAN message, respectively. If FSCH = b01, the filters and mask are defined as shown in this table.

Table 420. Standard ID filter element with range scheme (FSCH = b01)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH=b01		Reserved		RTRfilter	STD ID Filter2		Reserved		RTR mask	STD ID Filter1	

- If FSCH = b10, the filter scheme is based on two filters without masks. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base frame format (IDE = 0)
  2. (ID[*n*] = STD ID Filter1[*n*]) or (ID[*n*] = STD ID Filter2[*n*]) for each bit *n* from 0 to 10
  3. (RTR = RTR Filter1) or (RTR = RTR Filter2)

RTR and ID are the Remote Transmit Request bit and ID from a CAN message, respectively. If FSCH = b10, the filters are defined as shown in this table.

Table 421. Standard ID filter element with 2-filter scheme (FSCH = b10)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH=b10		Reserved		RTR Filter 2	STD ID Filter2		Reserved		RTR Filter 1	STD ID Filter1	

Each pair of ERFFEL*n* registers can store one extended filter element. The matching criteria are defined by ERFFEL*n*[FSCH] in this way:

- If FSCH = b00, the filter scheme is based on mask and filter. A CAN message matches an extended ID filter element only if these criteria are reached:
  1. CAN message is extended frame format (IDE = 1)
  2. (ID[*n*] = EXT ID filter [*n*]) or (EXT ID Mask[*n*] =0) for each bit *n* from 0 to 28
  3. (RTR = RTR Filter) or (RTR MASK = 0)

If FSCH = b00, the filters and masks are defined as shown in this table.

Table 422. Extended ID filter element with filter + mask scheme (FSCH = b00)

31	30	29	28	0
FSCH		RTR filter	EXT ID filter	
Reserved		RTR mask	EXT ID mask	

- If FSCH = b01, the filter scheme is based on range. A CAN message matches an extended ID filter element only if the following criteria are reached:
  1. CAN message is extended frame format (IDE = 1)
  2. ID ≥ EXT ID Filter1
  3. ID ≤ EXT ID Filter2
  4. (RTR = RTR Filter) or (RTR MASK = 0)

If FSCH = b01, the filters and masks are defined as shown in this table.

Table 423. Extended ID filter element with range scheme (FSCH = b01)

31	30	29	28	0
FSCH		RTR filter	EXT ID Filter2	
Reserved		RTR mask	EXT ID filter 1	

- If FSCH = b10, the filter scheme is based on two filters without masks. A CAN message matches an extended ID filter element only if these criteria are reached:
  1. CAN message is extended frame format (IDE = 1)
  2. (ID[*n*] = EXT ID Filter1[*n*]) or (ID[*n*] = EXT ID Filter2[*n*]) for each bit *n* from 0 to 28
  3. (RTR = RTR Filter1) or (RTR = RTR Filter2)

If FSCH = b10, the filters and masks are defined as shown in this table.

Table 424. Extended ID filter element with 2-filter scheme (FSCH = b10)

31	30	29	28	0
FSCH		RTR Filter2	EXT ID Filter2	
Reserved		RTR Filter1	EXT ID filter 1	

70.3.8.2 Enhanced Rx FIFO under DMA operation

The DMA feature is enabled by asserting both ERF[ERFEN] and MCR[DMA]. The DMA controller can read the received message by reading a message buffer structure at the Enhanced FIFO output port at the address range defined in Enhanced Rx FIFO structure.

NOTE

FlexCAN supports 32-bit access only for DMA transfers.

The CPU should not access the Enhanced FIFO output port address range during DMA operation, so that the FIFO engine operates properly. Before enabling MCR[DMA], the CPU must service Enhanced Rx FIFO status bits. Otherwise, these bits may show that the FIFO has data to be serviced, and mistakenly generate a DMA request. Before disabling MCR[DMA], the CPU must first clear the ERFUFW, ERFOVF, ERMWMI, and ERFDA fields in the ERFSR register, and then clear the Enhanced RX FIFO engine by writing one to ERFSR[ERFCLR].

ERFSR[ERFDA] is set by the hardware when there is one frame available to be read from the Enhanced Rx FIFO. Upon receiving the request, the DMA controller can read the message in the Enhanced Rx FIFO output. Each message reading process must end by the address defined in ERFCR[DMALW].

The following rules must be obeyed for Enhanced Rx FIFO DMA operation:

- Because a DMA transfer cannot be dynamically changed, ERFCR[DMALW] should be programmed so that the Enhanced Rx FIFO element can store the largest CAN message present on the CAN bus.
- Data bytes are valid according to the DLC field. See Table 449.

Each time one message is read from FIFO by the DMA controller, ERFSR[ERFDA] is cleared by FlexCAN and set again if there is at least one message stored in the FIFO.

For example, suppose that the maximum number of bytes in the data field of a CAN frame for a certain application is eight, and high-resolution time stamp is enabled. In that case the last Enhanced RX FIFO address offset can be found in Table 458 and Table 459. Using this address offset, ERFCR[DMALW] can be determined like this:

- Maximum number of data bytes = 8
- HR TIME STAMP enabled
- Last address offset = TS_OFF = 2014h
- DMALW = 5

70.3.8.3 Enhanced Rx FIFO clear operation

When ERFCR[ERFEN] is set, the CPU can clear the Enhanced Rx FIFO by writing one into ERFSR[ERFCLR]. The clear operation resets the internal FIFO pointers although the FIFO content stored in RAM is not changed. This operation can only be performed in Freeze mode and is blocked by hardware in other modes. This operation does not clear the ERFUFW, ERFOVF, ERFDA, and ERFWMI fields in EFRSR. Consequently the CPU must service all these fields before executing the clear FIFO task.

70.3.9 Legacy Rx FIFO

The Legacy Rx FIFO is receive-only and is enabled by asserting MCR[RFEN]. The reset value of this bit is zero to maintain software backward compatibility with previous versions of the module that did not have the Legacy FIFO feature.

CAUTION

Legacy Rx FIFO must not be enabled when CAN FD feature is enabled.

The Legacy FIFO is 6-message deep. The memory region occupied by the Legacy FIFO structure (both message buffers and Legacy FIFO engine) is described in Legacy Rx FIFO structure. The CPU can read the received messages sequentially, in the order they were received, by repeatedly reading a message buffer structure at the output of the Legacy FIFO.

IFLAG1[BUF5I] (Frames available in Legacy Rx FIFO) is asserted when there is at least one frame available to be read from the Legacy FIFO. An interrupt is generated if it is enabled by the corresponding mask bit. Upon receiving the interrupt, the CPU can read the message (accessing the output of the Legacy FIFO as a message buffer) and the RXFIR register, and then clear the interrupt. If there are more messages in the Legacy FIFO the act of clearing the interrupt updates the output of the Legacy FIFO with the next message and updates RXFIR with the attributes of that message, reissuing the interrupt to the CPU. Otherwise, the flag remains negated. The output of the Legacy FIFO is valid only while IFLAG1[BUF5I] is asserted.

IFLAG1[BUF6I] (Legacy Rx FIFO Warning) is asserted when the number of unread messages within the Legacy Rx FIFO is increased to five from four due to the reception of a new one, meaning that the Legacy Rx FIFO is almost full. The flag remains asserted until the CPU clears it.

IFLAG1[BUF7I] (Legacy Rx FIFO Overflow) is asserted when an incoming message was lost because the Legacy Rx FIFO is full. Note that the flag will not be asserted when the Legacy Rx FIFO is full and the message was captured by a Mailbox. The flag remains asserted until the CPU clears it.

Clearing one of those three flags does not affect the state of the other two.

An interrupt is generated if an IFLAG bit is asserted and the corresponding mask bit is asserted too.

A powerful filtering scheme is provided to accept only frames intended for the target application, reducing the interrupt servicing workload. The filtering criteria is specified by programming a table of up to 128 32-bit registers, according to CTRL2[RFFN] setting, that can be configured to one of the following formats (see also Legacy Rx FIFO structure):

- Format A: 128 IDAFs (extended or standard IDs including IDE and RTR)
- Format B: 256 IDAFs (standard IDs or extended 14-bit ID slices including IDE and RTR)
- Format C: 512 IDAFs (standard or extended 8-bit ID slices)

NOTE

A chosen format is applied to all entries of the filter table. It is not possible to mix formats within the table.

Every frame available in the Legacy Rx FIFO has a corresponding IDHIT (Identifier Acceptance Filter Hit Indicator) that can be read in the IDHIT field from C/S word, as shown in the Legacy Rx FIFO Structure description. Another way the CPU can obtain this information is by accessing the RXFIR register. RXFIR[IDHIT] refers to the message at the output of the Legacy FIFO and is valid while the IFLAG1[BUF5I] flag is asserted. The RXFIR register must be read only before clearing the flag, which guarantees that the information refers to the correct frame within the Legacy FIFO.

Up to 32 elements of the filter table are individually affected by the Individual Mask Registers (RXIMRx), according to the setting of CTRL2[RFFN], allowing very powerful filtering criteria to be defined. If MCR[IRMQ] is negated, then the Legacy Rx FIFO filter table is affected by RXFGMASK.

NOTE

For more information about the difference between FD and non-FD regarding this feature, see Table 414.

70.3.9.1 Legacy Rx FIFO under DMA operation

The receive-only Legacy FIFO can support DMA. This feature is enabled by asserting both MCR[RFEN] and MCR[DMA]. The reset value of MCR[DMA] is zero to maintain backward compatibility with previous versions of the module that did not have the DMA feature.

The DMA controller can read the received message by reading a message buffer structure at the Legacy FIFO output port at the 80h–8Ch address range.

NOTE

FlexCAN supports 32-bit access only for DMA transfers.

When MCR[DMA] is asserted the CPU must not access the Legacy FIFO output port address range. Before enabling MCR[DMA], the CPU must service the IFLAGs asserted in the Legacy Rx FIFO region. Otherwise, these IFLAGs may show that the FIFO has data to be serviced, and mistakenly generate a DMA request. Before disabling MCR[DMA], the CPU must perform a clear Legacy FIFO operation.

IFLAG1[BUF5I] (Frames available in Legacy Rx FIFO) is asserted when there is at least one frame available to be read from the FIFO. Consequently a DMA request is generated simultaneously. Upon receiving the request, the DMA controller can read the message (accessing the output of the Legacy FIFO as a message buffer). The DMA reading process must end by reading address 8Ch, which clears IFLAG1[BUF5I] and updates both the FIFO output with the next message (if FIFO is not empty) and the RXFIR register with the attributes of the new message. If there are more messages stored in the FIFO, IFLAG1[BUF5I] will be re-asserted and another DMA request is issued. Otherwise, the flag remains negated.

NOTE

RXFIR register contents cannot be read after DMA completes the Legacy FIFO read. The IDHIT information is also available in the C/S word at address 080h (see Legacy Rx FIFO structure.

IFLAG1[BUF6I] and IFLAG1[BUF7I] are not used when the DMA feature is enabled.

When FlexCAN is working with DMA, the CPU does not receive any Legacy Rx FIFO interruption and must not clear the related IFLAGs. In addition, the related IMASKs are not used to mask the generation of DMA requests.

NOTE

For more information about the difference between FD and non-FD regarding this feature, see Table 414.

70.3.9.2 Clear Legacy FIFO operation

When MCR[RFEN] is asserted, the clear Legacy FIFO operation is a feature used to empty Legacy FIFO contents. With MCR[RFEN] asserted the Clear FIFO occurs when the CPU writes one in IFLAG1[BUF0I]. This operation can only be performed in Freeze mode and is blocked by hardware in other modes. This operation does not clear the FIFO IFLAGs; consequently the CPU must service all FIFO IFLAGs before executing the clear FIFO task.

When Legacy Rx FIFO is working with DMA, the clear FIFO operation clears IFLAG1[BUF5I] and the DMA request is canceled.

CAUTION

Clear Legacy FIFO operation does not clear IFLAGs, except when MCR[DMA] is asserted; in this case only IFLAG1[BUF5I] is cleared.

70.3.10 CAN protocol related features

This section describes the CAN protocol related features.

70.3.10.1 CAN FD ISO compliance

The CAN FD protocol has been improved to increase the failure detection capability that was in the original CAN FD protocol, which is also called non-ISO CAN FD, by CAN in Automation (CiA). A three-bit stuff counter and a parity bit have been introduced in the improved CAN FD protocol, now called ISO CAN FD. The CRC calculation has also been modified. All these improvements make the ISO CAN FD protocol incompatible with the non-FD CAN FD protocol. The non-ISO CAN FD is still supported by FlexCAN so that it can be used mainly during an intermediate phase, for evaluation and development purposes.

Therefore, it is strongly recommended to configure FlexCAN to the ISO CAN FD protocol by setting the ISOCANFDEN field in the CTRL2 register.

70.3.10.2 CAN FD frames

The ISO 11898-1 standard specifies the Classical Frame format compliant to ISO 11898-1 (2003) and introduces the CAN Flexible Data Rate Frame format. The Classical Frame format allows bit rates up to 1 Mbit/s and payloads up to 8 bytes per frame. The Flexible Data Rate Frame format allows bit rates higher than 1 Mbit/s and payloads longer than 8 bytes per frame. FlexCAN can receive and transmit CAN FD messages interleaved with Classical CAN messages.

There are three additional control bits in the CAN FD frame. The Extended Data Length (EDL) bit enables a longer data payload with different data length coding. The Bit Rate Switch (BRS) bit decides whether the bit rate is switched inside a CAN FD format frame. The Error State Indicator (ESI) flag is transmitted dominant by error active nodes, and recessive by error passive nodes. There are no Remote Frames (see Remote frames) in the CAN FD format. A message configured to transmit a Remote Frame is always sent out in the Classical CAN format. When an FD frame is received and matches a mailbox, the RTR bit in the receiving message buffer is negated. The RTR bit must be considered in classical frames only.

CAN FD messages may be formatted as long frames where the data field exceeds 8 bytes, and may range from 12 up to 64 bytes. They can also be configured to support bit rate switching, where the control field, the data field, and the CRC field of a CAN frame are transmitted with a higher bit rate than the beginning and the end of the frame. Messages in Classical CAN format are limited to transport a maximum payload of 8 bytes at nominal rate. The following figure illustrates the message formats for Classical and FD frames with either standard or extended ID.

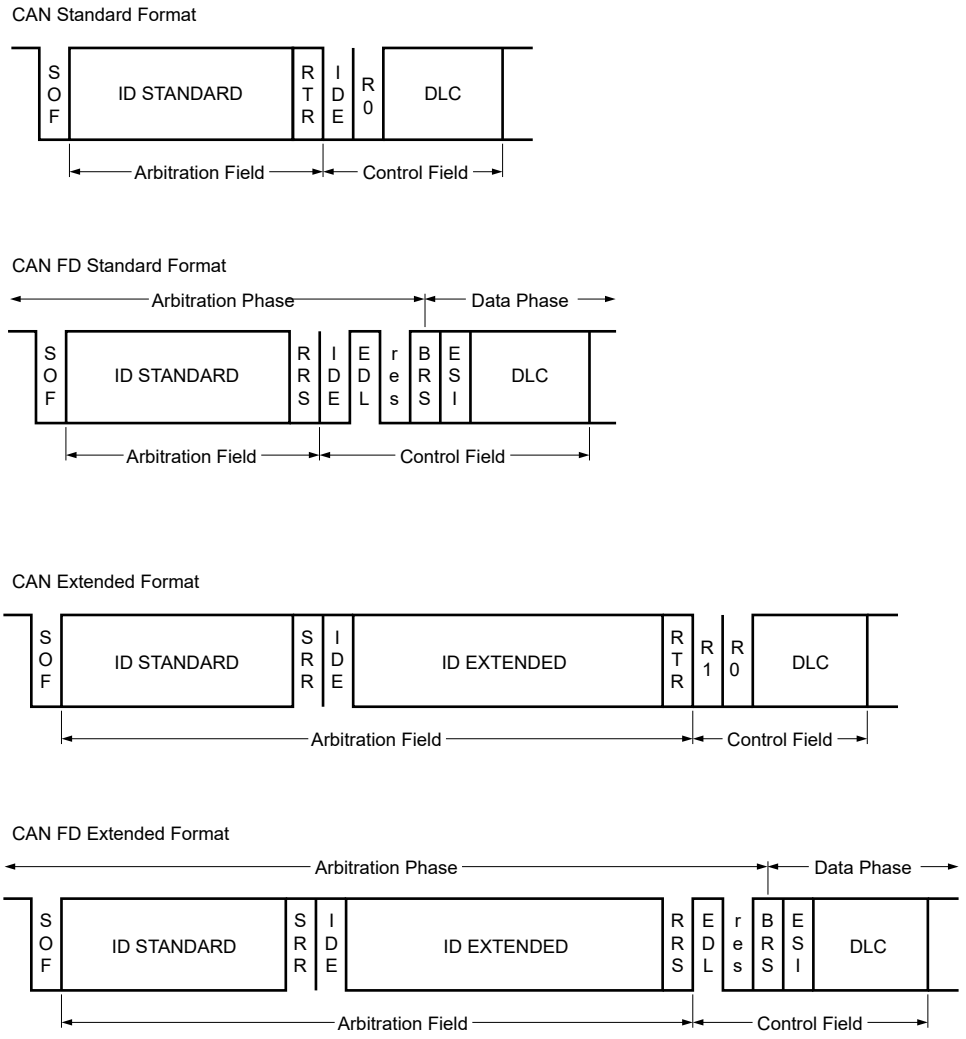


Figure 374. CAN message formats

The ability to receive and transmit CAN FD messages is enabled by MCR[FDEN]. Either a recessive R0 bit in CAN frames with 11-bit identifiers or a recessive R1 bit in CAN frames with 29-bit identifiers are decoded as an EDL bit (not a reserved one). A CAN FD frame is recognized by a recessive EDL bit, and a Classical CAN frame is recognized by a dominant EDL bit. The BRS bit specifies whether this frame switches the bit rate in its data phase. A long frame is decoded in accordance with the DLC field value (see DLC definition in Message buffer structure).

CAN FD messages can be transmitted with two different bit rates. The first part of a CAN FD frame, from the Start Of Frame (SOF) bit until the Bit Rate Switch (BRS) bit, also called the arbitration phase, is transmitted with the nominal bit rate based on a set of nominal CAN bit timing configuration values. The second part, from the BRS bit until the CRC Delimiter bit, also named the data phase, is transmitted with the data bit rate defined by a second set of CAN data bit timing configuration values. Finally, from the CRC Delimiter until the Intermission bits, the transmission returns to nominal bit rate. In CAN FD frames with bit rate switching, the bit timing is changed inside the frame at the sample point of the BRS bit if this bit is recessive. Before the BRS bit, in the CAN FD arbitration phase, the nominal CAN bit timing is used as defined by the CBT register (also by CTRL1 register for backward compatibility). Upon detecting a recessive BRS bit, the CAN data bit timing is used as defined by the FDCBT register.

NOTE

If the length of the time quantum in the nominal bit timing and the length of the time quantum in the data bit timing are not identical, a quantization error of up to one time quantum of the arbitration phase may be present as a phase error. This situation can occur after the switch from arbitration to data phase and will last until the next synchronization event. Thus, the length of the time quantum should be the same in nominal and data bit timing in order to minimize the chance of error frames on the CAN bus, and to optimize the clock tolerance in networks that use FD frames.

FDCTRL[FDRATE] enables the transmission of all frames with bit rate switching if the BRS bit in the selected Tx MB is set. If FDRATE is negated, the transmission is performed at nominal rate regardless of the BRS bit value. FDCTRL[FDRATE] can be written any time but takes effect only for the next message transmitted or received.

The nominal bit timing is resumed at either the sample point of the CRC Delimiter bit or when an error is detected, whichever occurs first. The following figure describes the mechanism for entering and leaving the data phase when BRS bit is recessive.

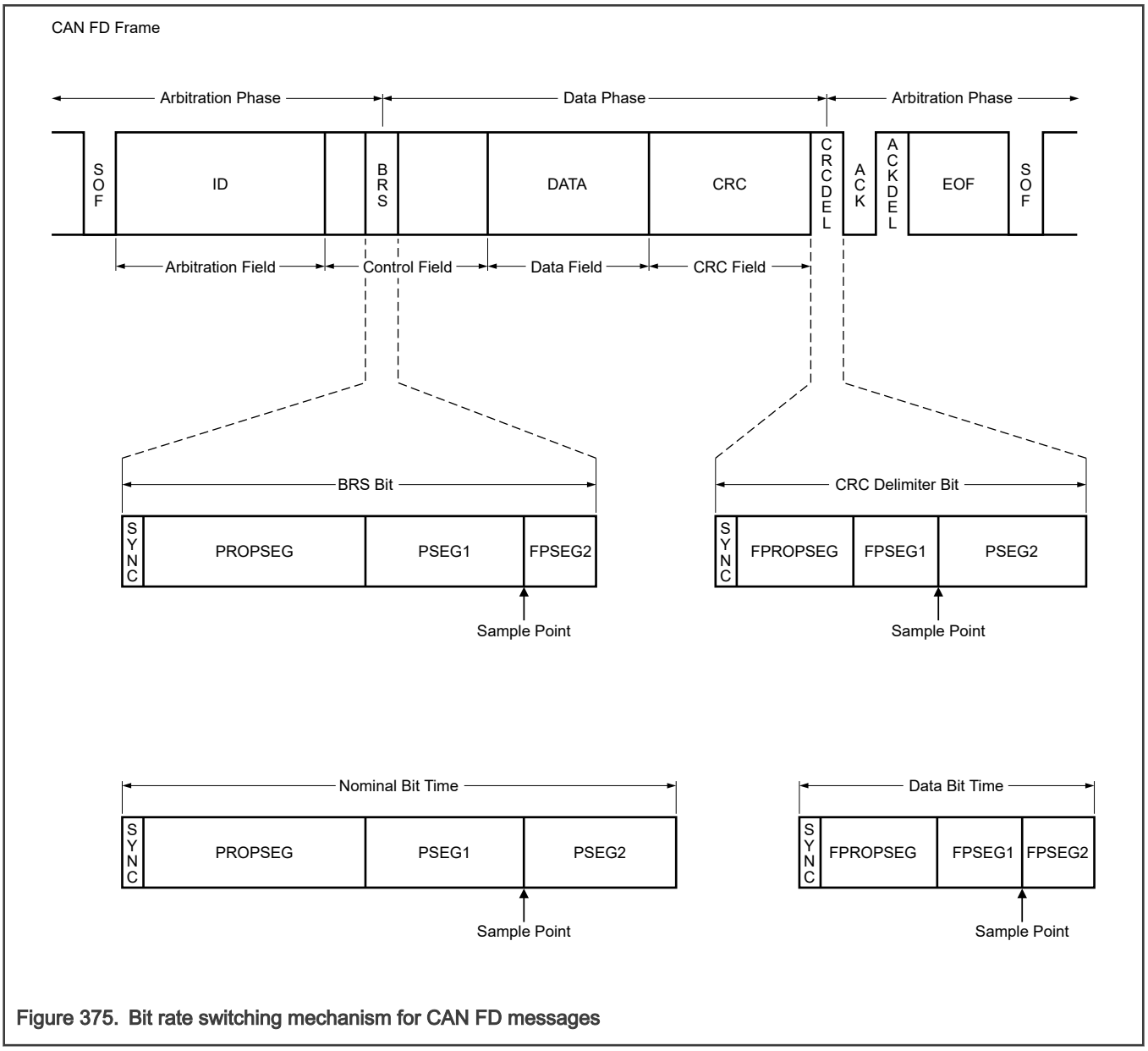


Figure 375. Bit rate switching mechanism for CAN FD messages

NOTE

In Classical CAN frames, the CRC delimiter is one single recessive bit. In CAN FD frames, the CRC delimiter may consist of one or two recessive bits. FlexCAN sends only one recessive bit as the CRC delimiter, but it accepts two recessive bits before the edge from recessive to dominant that starts the acknowledge slot. As a receiver, FlexCAN sends its acknowledge bit after the first CRC delimiter bit. In CAN FD frames, FlexCAN accepts a two-bit dominant ACK slot as a valid ACK to compensate for phase shifts between the receivers.

The maximum configurable bit rate in the CAN FD data phase depends on the clock frequency of CAN_PE subblock. For example, with a CAN_PE clock frequency of 40 MHz and the shortest configurable bit time of 5 time quanta, the bit rate in the data phase is 8 Mbit/s.[11]

The value of the ESI bit is determined either by the transmitter's error state at the start of the transmission, if the frame is originated in the FlexCAN node, or by the original transmitting node in case FlexCAN is acting as a gateway for the message. If the transmitter is error passive, ESI is transmitted recessive; otherwise, it is transmitted dominant. The permutations of the relationship between the written value and the transmitted value of the ESI are shown in this table.

Table 425. Written versus transmitted values of ESI field

FlexCAN fault confinement status at start of frame	ESI bit Of Tx MB	Transmitted ESI
Error active	0	0 (Error Active)
Error passive	0	1 (Error Passive)
Error active	1	1 (Error Passive)
Error passive	1	1 (Error Passive)

There are different CRC polynomials for different CAN frame formats. The first polynomial, CRC_15, is used for all frames in Classical CAN format. The second, CRC_17, is used for frames in CAN FD format with a data field up to sixteen bytes long. The third, CRC_21, is used for frames in CAN FD format with a data field longer than sixteen bytes. Each polynomial results in a Hamming distance of 6. At the start of the frame, all three CRC polynomials are calculated concurrently. The CRC sequence to be transmitted is selected by the values of the EDL bit and the DLC bit field. When receiving a message, FlexCAN decodes EDL and DLC to select the adequate CRC polynomial to check for a CRC error.

In CAN FD format frames, stuff bits are included in the bit stream for CRC calculation. In Classical CAN format frames, stuff bits are not included. After the transmission of the last bit relevant to the CRC calculation, the FDCRC register stores the calculated CRC for the transmitted message, with the adequate length in accordance to the type of message, for both CAN FD and non-FD messages. The CRCR register reports a valid CRC for Classical CAN messages only.

In CAN FD format frames, the CAN bit stuffing method is changed for the CRC sequence so that the stuff bits are inserted at fixed positions. When FlexCAN is transmitting a CAN FD frame, a fixed stuff bit is inserted just before the first bit of the CRC sequence, even if the last bits of the preceding field do not fulfill the CAN stuff condition. Additional stuff bits are inserted after each fourth bit of the CRC sequence. The value of any fixed stuff bit is the inverse value of its preceding bit. When FlexCAN is receiving a CAN FD frame, it discards the fixed stuff bits from the bit stream for the CRC check. A stuff error is detected if the fixed stuff bit has the same value as its preceding bit.

FlexCAN detects errors in CAN FD frames the same way as in Classical CAN frames. The error counters RXERRCNT and TXERRCNT in the ECR register accumulate the counts of Rx and Tx errors, respectively, for both FD and non-FD frames indiscriminately. There are two extra error counters (RXERRCNT_FAST and TXERRCNT_FAST) that accumulate Rx and Tx errors occurring in the data phase of CAN FD frames with the BRS bit set only. The rules for updating the error counters are the same for both CAN FD and non-FD frames (see ECR register).

Error Flags BITERR1, BITERR0, ACKERR, CRCERR, FRMERR, and STFERR in the ESR1 register report errors in both CAN FD and non-FD frames. They also generate the ERRINT interrupt if CTRL1[ERRMSK] is asserted. The ESR1 register has additional error flags (BITERR1_FAST, BITERR0_FAST, CRCERR_FAST, FRMERR_FAST, and STFERR_FAST) to individually indicate

[11] The frequency used in this example may not be supported on this chip; it is shown only to demonstrate how the maximum configurable bit rate is calculated.

the occurrence of errors in the data phase of CAN FD frames with the BRS bit set. There is no ACKERR detected in the data phase of a CAN FD frame. Fault confinement status reported in ESR1[FLTCONF] is the same for both CAN FD and Classical CAN frames, and is based on RXERRCNT and TXERRCNT error counters only. Information contained in RXERRCNT_FAST and TXERRCNT_FAST counters may be considered as status to help detect the error nature related to the bit rate value.

When FlexCAN is in the data phase, either transmitting or receiving a CAN FD message, and detects an error, it immediately switches back to the arbitration phase and to the nominal rate to start an error flag.

Resynchronization and hard synchronization occur in CAN FD frames in the same way as in Classical CAN ones. Additionally, a hard synchronization is also performed at the recessive to dominant edge from EDL to R0 in CAN FD format frames. FlexCAN does not resynchronize while transmitting in the CAN FD data phase.

70.3.10.3 Transceiver delay compensation

The CAN FD protocol allows the transmission and reception of data at a higher bit rate than the nominal rate used in the arbitration phase when the message's BRS bit is set. This feature enables the use of rates up to 8 Mbps.

During the data phase of a CAN FD frame, the transmitter detects a bit error if it cannot receive its own latest transmitted bit at the sample point of that bit. When bit rate switching is enabled (BRS bit is asserted), the length of the CAN bit time in the data phase can become shorter than the transceiver's loop delay, thus impeding the correct comparison between the transmitted bit and the received bit within the current CAN bit time interval.

Note that the TDC process defines a secondary sample point where the transmitted bit is correctly compared with the received bit in order to check for bit errors.

The TDC mechanism can be enabled by FDCTRL[TDCEN] or ETDC[ETDCEN] and is effective only during the data phase of FD frames having the BRS bit set. It has no effect either on non-FD frames, or on FD frames transmitted at normal bit rate. The TDC is active from the sample point of the BRS bit until the sample point of the CRC Delimiter bit, provided the respective message under transmission has the BRS bit set. When it is active, a comparison is done between the real received bit and the delayed transmitted bit, where the delay is calculated based on the measured transceiver loop delay.

NOTE

The actual value of the CRC Delimiter bit is disregarded by transmitters using the transceiver delay compensation mechanism. A global error at the end of the CRC field will cause the receivers to send error frames that the transmitter will detect during Acknowledge or End of Frame.

For every transmitted FD frame having the BRS bit set, the delay measurement is triggered by the transition from the recessive EDL bit to the dominant R0 bit (as shown in the next figure). The loop delay is measured in Protocol Engine (PE) clock periods (CANCLK, see Protocol timing), from the transmitted EDL-R0 edge to the received EDL-R0 edge. The position of the secondary sample point is defined by the measured loop delay time added to an offset value specified in FDCTRL[TDCOFF] or ETDC[ETDCOFF]. FDCTRL[TDCVAL] or ETDC[ETDCVAL] stores the result of this calculation. The TDCVAL and ETDCVAL value saturates at its maximum value of 63 CANCLK and 255 CANCLK when the delay measurement is too long.

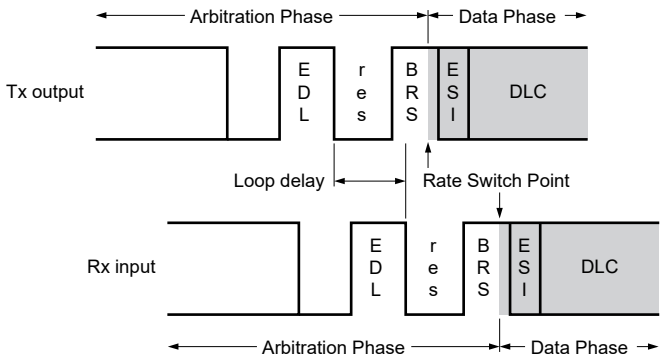


Figure 376. Transceiver loop delay measurement

The measured loop delay is not enough to be used to define the secondary sample point because it relates to the CAN bit edges. The transceiver delay compensation offset TDCOFF or ETDCOFF is used to shift the secondary sample point from the edge to an intermediate point inside the bit time, far away from its edges. Therefore, the TDCOFF or ETDCOFF value cannot be larger than the CAN bit duration in the data phase.

If the secondary sample point is set very near the CAN bit edge (SYNC field), then problems may occur during the bit sampling in the data phase. For the TDC to work reliably, the offset has to use optimal settings. To be sure the bit sampling is performed in the best region, the TDC offset should be configured as shown in this equation:

$$\text{Offset} = (\text{FPSEG1} + \text{FPROPSEG} + 2) \times (\text{FPRESDIV} + 1)$$
or
$$\text{Offset} = (\text{DTSEG1} + 2) \times (\text{EDPRESDIV} + 1), \text{ if ETDCEN}$$

The following figure shows the SSP position when these settings are used.

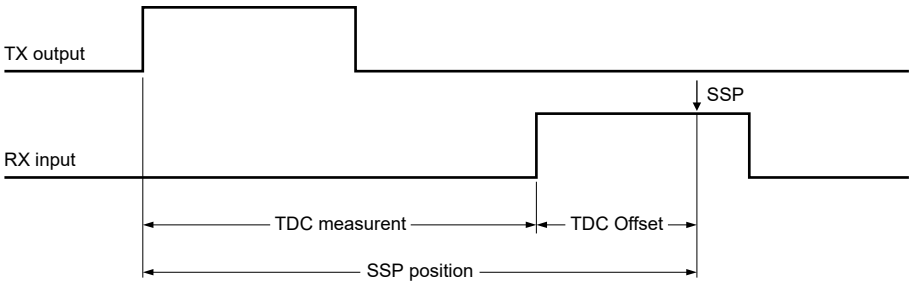


Figure 377. SSP position with optimal values

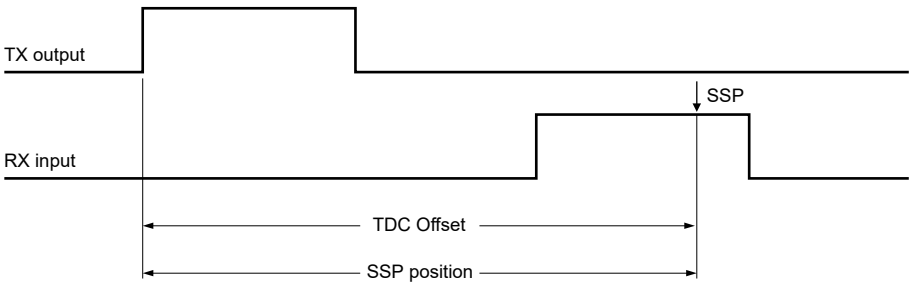


Figure 378. SSP position—TDC measurement disabled by ETDCEN and TDMDIS

Alternatively, if CTRL2[BTE] and ETDC[ETDCEN] are set, the TDMDIS bit can be set to disable the transceiver delay measurement. In this case, the SSP position is defined only by the ETDCOFF field. Figure 378 shows the secondary sample point position when the transceiver delay measurement is disabled.

During the data phase of CAN FD frames with bit rate switching enabled, at the onset of every Tx CAN bit, the transmitted Tx bit value is temporarily stored in a buffer and a time countdown based on FDCTRL[TDCVAL] or ETDC[ETDCVAL] is started which ends with the comparison of the received Rx bit (delayed by the external loop delay plus the specified offset) with the stored Tx bit. If a bit error is detected at the secondary sample point, the FlexCAN issues an error flag to the CAN bus at the next sample point.

During the arbitration phase the delay compensation is always disabled. The maximum delay which can be compensated by the FlexCAN's transceiver delay compensation during the data phase is 3 CAN bit times – 2 Tq. Beyond this limit, the FDCTRL[TDCFAIL] or ETDC[ETDCFAIL] flag is set to indicate when the transceiver delay compensation mechanism is out of range, unable to compensate the transceiver loop delay.

70.3.10.4 Remote frames

A remote frame is a special kind of frame. You can program a mailbox to be a remote request frame by configuring the mailbox as Transmit with the RTR bit set to one. After the remote request frame is transmitted successfully, the mailbox becomes a receive message buffer, with the same ID as before.

When a remote request frame is received by FlexCAN, it can be treated in four ways, depending on remote request storing (CTRL2[RRS]) and Rx FIFO Enable (MCR[RFEN]):

- If RRS is negated the frame's ID is compared to the IDs of the transmit message buffers with the CODE field 1010b. If there is a matching ID, then this mailbox frame will be transmitted. Note that if the matching mailbox has the RTR bit set, then FlexCAN will transmit a remote frame as a response. The received remote request frame is not stored in a receive buffer. It is only used to trigger a transmission of a frame in response. The mask registers are not used in remote frame matching, and all ID bits (except RTR) of the incoming received frame should match. In the case that a remote request frame is received and matches a mailbox, this message buffer immediately enters the internal arbitration process, but is considered as a normal Tx mailbox, with no higher priority. The data length of this frame is independent of the DLC field in the remote frame that initiated its transmission.
- If RRS is asserted the frame's ID is compared to the IDs of the receive mailboxes with the CODE field 0100b, 0010b, or 0110b. If there is a matching ID, then this mailbox will store the remote frame in the same fashion of a data frame. No automatic remote response frame will be generated. The mask registers are used in the matching process.
- If RFEN is asserted FlexCAN will not generate an automatic response for remote request frames that match the Legacy FIFO filtering criteria. If the remote frame matches one of the target IDs, it will be stored in the Legacy FIFO and presented to the CPU. Note that for filtering formats A and B, it is possible to select whether remote frames are accepted or not. For format C, remote frames are always accepted (if they match the ID). Remote request frames are considered as normal frames, and generate a Legacy FIFO overflow when a successful reception occurs and the Legacy FIFO is already full.
- If ERFEN is set, FlexCAN will not generate an automatic response for remote request frames that match the Enhanced Rx FIFO filtering criteria. Remote Request Frames are considered normal frames, and generate an Enhanced Rx FIFO overflow when a sucessful reception occurs and Enhanced Rx FIFO is already full.

NOTE

There is no remote frame in the CAN FD format. The RTR bit is replaced by a fixed dominant RRS bit. FlexCAN receives and transmits remote frames in the Classical CAN format.

70.3.10.5 Overload frames

FlexCAN does transmit overload frames when the following conditions are detected on the CAN bus:

- Detection of a dominant bit in the first/second bit of Intermission
- Detection of a dominant bit at the 7th bit (last) of End of Frame field (Rx frames)
- Detection of a dominant bit at the 8th bit (last) of Error Frame Delimiter or Overload Frame Delimiter

70.3.10.6 Message buffer time stamp

The value of the free running timer is sampled at the beginning of the Identifier field on the CAN bus, and is stored at the end of move-in in the TIME STAMP field of message buffer, providing network behavior with respect to time.

When CTRL2[TIMER_SRC] is asserted, the free running timer is continuously clocked by an external time tick.

When CTRL2[TIMER_SRC] is negated, the free running timer is clocked by the FlexCAN bit-clock, which defines the baud rate on the CAN bus. During a message transmission/reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate.

The free running timer is not incremented during Disable and Freeze modes. It can be reset upon a specific frame reception, enabling network time synchronization. See the TSYN description in Control 1 register (CTRL1).

Alternatively, by configuring the MTSBASE field of the Control 2 register (CTRL2), the time stamp of the message buffer can capture the lower or higher 16 bits of the high-resolution dedicated counter.

70.3.10.7 High resolution time stamp

The high-resolution time stamp, HR_TIME_STAMP, uses a dedicated timer with a 32-bit counter operating in free running mode. The high-resolution time stamp is enabled by the TSTAMPCAP field of the Control 2 register (CTRL2). When this field is not zero , the dedicated 32-bit counter value is captured during a valid CAN frame and stored in the HR_TIME_STAMP register.

Each HR_TIME_STAMP*n* corresponds to a specific message buffer. For example, HR_TIME_STAMP0 stores the 32-bit time stamp associated with message buffer 0, HR_TIME_STAMP1 stores the 32-bit time stamp associated with message buffer 1, and so on.

The counter value is captured according to the TSTAMPCAP field of the Control 2 register (CTRL2). For classical CAN frames, the capture points can be the start of frame bit or the point in time a CAN frame is considered valid, which is the seventh bit of end of frame for transmission and the sixth bit of end of frame for reception. For CAN FD frames, the capture points can be the start of frame, the point in time a CAN FD frame is considered valid, or the res bit of a CAN FD frame.

The 16-bit time stamp of the message buffer can be configured to capture the lower or higher 16 bits of the high-resolution timer. This configuration is made by CTRL2[MBTSBASE].

70.3.10.8 Protocol timing

The following figure shows the structure of the clock generation circuitry that feeds the CAN Protocol Engine (PE) submodule.

NOTE

Please see the clock distribution chapter (module clocks table) to identify the proper clock source.

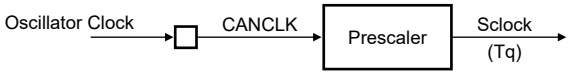


Figure 379. CAN engine clocking scheme

The FlexCAN module supports a variety of means to set up bit timing parameters that are required by the CAN protocol. The Control 1 register (CTRL1) has various fields used to control bit timing parameters: PRESDIV, PROPSEG, PSEG1, PSEG2, and RJW.

The CAN Bit Timing register (CBT) extends the range of the CAN bit timing variables in CTRL1. The CAN FD Bit Timing register (FDCBT) provides a second set of CAN bit timing variables to be applied at the data phase of CAN FD frames with the Bit Rate Switch (BRS) set.

The Enhanced Nominal CAN bit timing register (ENCBT) extends the range of CAN bit timing variables in CBT. The Enhanced Data Phase CAN bit timing register (EDCBT) extends the range of CAN bit timing variables in FDCBT. When using ENCBT and EDCBT, the bit timing nominal and data phase serial clock (Sclock) dividers must be programed in the Enhanced CAN Bit Timing Prescalers register (EPRS).

NOTE

When the CAN FD feature is enabled, always set CBT[BTF] or CTRL2[BTE] and configure the CAN bit timing variables in CBT or ENCBT. See CAN Bit Timing Register (CBT) or Enhanced Nominal CAN Bit Timing (ENCBT).

The PRESDIV field (as well as its extended range EPRESDIV (or ENPRESDIV) and FDPRESDIV (or EDPRESDIV) for the data phase bits of CAN FD messages) defines the prescaler value (see the equation below) that generates the serial clock (Sclock), whose period defines the time quantum used to compose the CAN waveform. A time quantum (Tq) is the atomic unit of time managed by the CAN engine.

$$Tq = \frac{(PRESDIV + 1)}{f_{CANCLK}}$$

Equation 10. Time quantum

The bit rate, which defines the rate the CAN message is either received or transmitted, is given by the formula:

$$\text{CAN Bit Time} = (\text{Number of Time Quanta in 1 bit time}) * \text{Tq}$$

$$\text{Bit Rate} = \frac{1}{\text{CAN Bit Time}}$$

Equation 11. CAN bit time and baud rate

A bit time is subdivided into three segments[1] (see Figure 381, Figure 382 and Table 426):

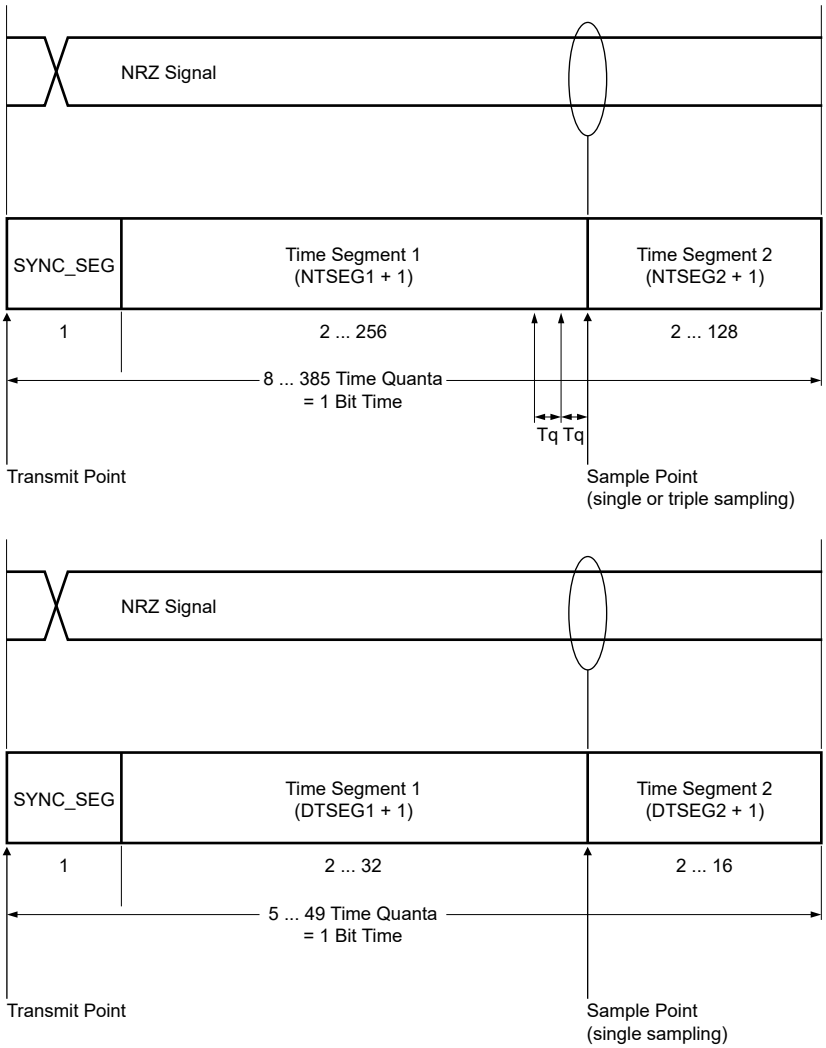


Figure 380. Segments within the bit timing (example using ENCBT and EDCBT bit timing variables)

- SYNC_SEG: This segment has a fixed length of one time quantum. Signal edges are expected to happen within this section.
- Time Segment 1: This segment includes the propagation segment and the phase segment 1 of the CAN standard. It can be programmed by setting CTRL1[PROPSEG] and CTRL1[PSEG1] so that their sum (plus 2) is in the range of 2 to 16 time quanta. When CBT[BTF] is asserted, FlexCAN uses EPROPSEG and EPSEG1 fields from CBT register so that their

[1] For further explanation of the underlying concepts, see ISO 11898-1. See also the CAN 2.0A/B protocol specification for bit timing.

sum (plus 2) is in the range of 2 to 96 time quanta. For messages in CAN FD format with the BRS bit set, FlexCAN uses FDPROPSEG and FDPSEG1 from FDCBT instead, so that their sum (plus 1) is in the range of 2 to 39 time quanta.

If CTRL2[BTE] is set, FlexCAN uses ENCBT[NTSEG1] to configure time segment 1 in the range of 2 to 256 time quanta. For the data phase in CAN FD messages with BRS set, EDCBT[DTSEG1] must be used for configuring time segment 1 in the range of 2 to 32 time quanta.

- Time Segment 2: This segment represents the phase segment 2 of the CAN standard. It can be programmed by setting CTRL1[PSEG2] (plus 1) to be 2 to 8 time quanta long. When CBT[BTF] is asserted, FlexCAN uses EPSEG2 fields of CBT register so that its value (plus 1) is in the range of 2 to 32 time quanta. For messages in CAN FD format with the BRS bit set, FlexCAN uses FDPSEG2 from FDCBT instead, so that its value (plus 1) is in the range of 2 to 8 time quanta. The time segment 2 cannot be smaller than the Information Processing Time (IPT), which value is 2 time quanta in FlexCAN.

  If CTRL2[BTE] is set, FlexCAN uses ENCBT[NTSEG2] to configure time segment 2 in the range of 2 to 128 time quanta. For the data phase in CAN FD messages with BRS set, EDCBT[DTSEG2] must be used for configuring time segment 2 in the range of 2 to 16 time quanta.

NOTE

The bit time defined by the above time segments must not be smaller than five time quanta. For bit time calculations, use an Information Processing Time (IPT) of two, which is the value implemented in the FlexCAN module.

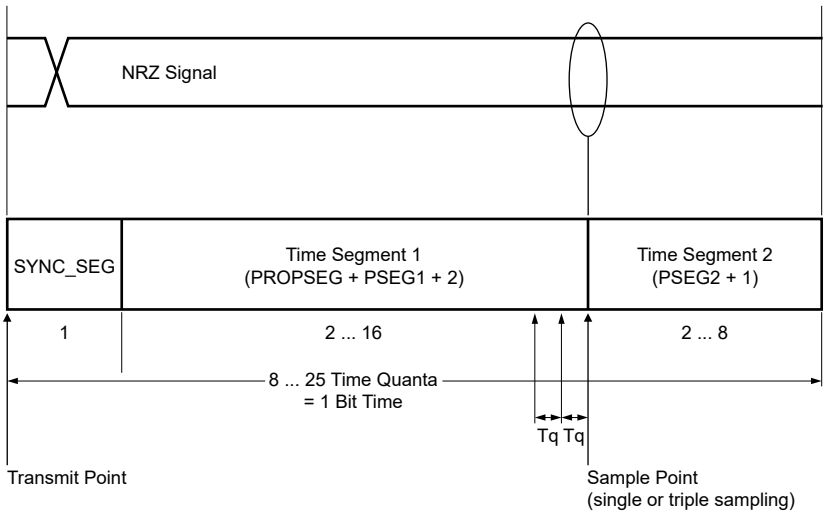


Figure 381. Segments within the bit time (example using CTRL1 bit timing variables for Classical CAN format)

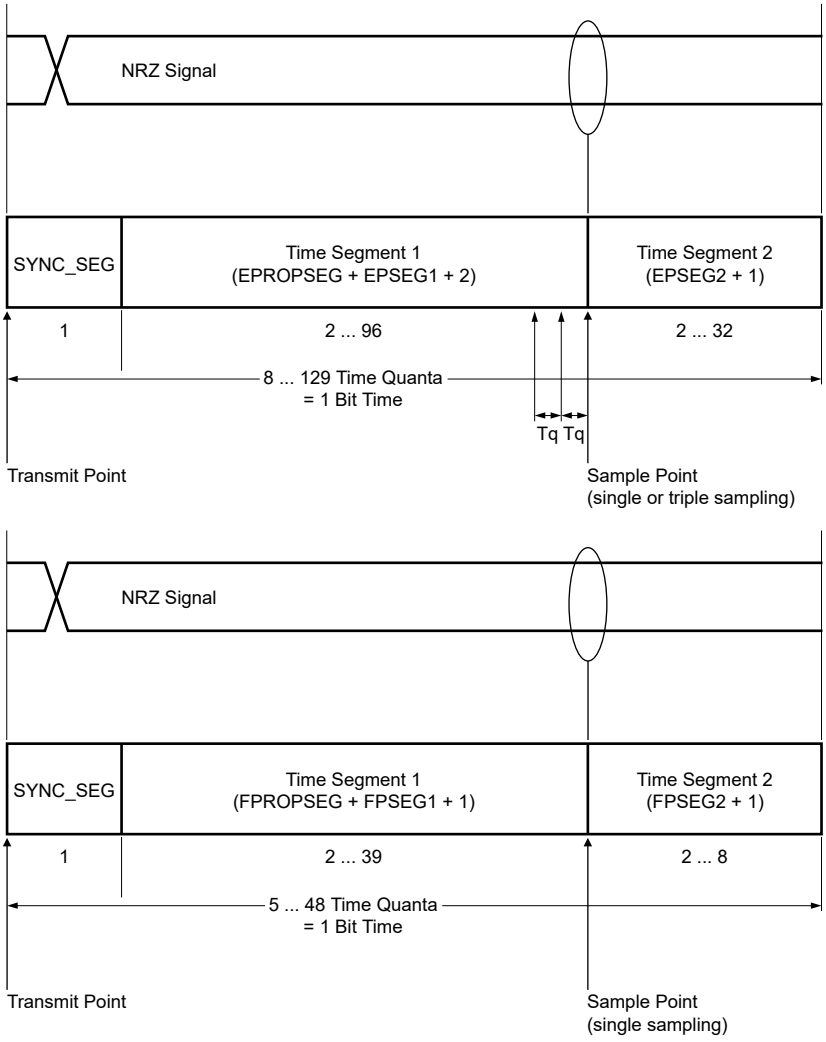


Figure 382. Segments within the bit time (example using CBT and FDCBT bit timing variables for CAN FD format)

Table 426. Time segment syntax

Syntax	Description
SYNC_SEG	System expects transitions to occur on the bus during this period.
TSEG1	Corresponds to the sum of PROPSEG and PSEG1.
TSEG2	Corresponds to the PSEG2 value.
Transmit point	A node in transmit mode transfers a new value to the CAN bus at this point.
Sample point	A node samples the bus at this point. If the three samples per bit option is selected, then this point marks the position of the third sample.

The following table gives some examples of the CAN compliant segment settings for Classical CAN format (Bosch CAN 2.0B) (non-FD) messages.

Table 427. Bosch CAN 2.0B standard compliant bit time segment settings

Time segment 1	Time segment 2	Re-synchronization jump width
5 .. 10	2	1 .. 2
4 .. 11	3	1 .. 3
5 .. 12	4	1 .. 4
6 .. 13	5	1 .. 4
7 .. 14	6	1 .. 4
8 .. 15	7	1 .. 4
9 .. 16	8	1 .. 4

NOTE

The user must ensure the bit time settings are in compliance with the CAN Protocol standard (ISO 11898-1).

Whenever CAN bit is used as a measure of time duration (for example, estimating the occurrence of a CAN bit event in a message), the number of peripheral clocks in one CAN bit (NumClkBit) can be calculated as:

$$NumClkBit = \frac{f_{SYS}}{f_{CANCLK}} \times (PRESDIV + 1) \times (PROPSEG + PSEG1 + PSEG2 + 4)$$

Equation 12. Number of peripheral clocks per CAN bit when CTRL2[BTE] = 0

or, if CTRL2[BTE] = 1:

$$NumClkBit = \frac{f_{SYS}}{f_{CANCLK}} \times (ENPRESDIV + 1) \times (NTSEG1 + NTSEG2 + 3)$$

Equation 13. Number of peripheral clocks per CAN bit when CTRL2[BTE] = 1

where:

- NumClkBit is the number of peripheral clocks in one CAN bit.
- f_{CANCLK} is the Protocol Engine (PE) Clock (see Figure 379), in Hz.
- f_{SYS} is the frequency of operation of the system (CHI) clock, in Hz.
- PSEG1 is the value in CTRL1[PSEG1] field.
- PSEG2 is the value in CTRL1[PSEG2] field.
- PROPSEG is the value in CTRL1[PROPSEG] field.
- PRESDIV is the value in CTRL1[PRESDIV] field.
- ENPRESDIV is the value in EPRS[ENPRESDIV] field.
- NTSEG1 is the value in ENCBT[NTSEG1] field.
- NTSEG2 is the value in ENCBT[NTSEG2] field.

The formula above is also applicable to the alternative CAN bit timing variables described in the CAN Bit Timing register (CBT) or the Enhanced Nominal CAN bit timing (ENCBT) register and also to the CAN FD Bit Timing register (FDCBT) or the Enhanced Nominal CAN bit timing register(ENCBT).

For example, 180 CAN bits = (180 x NumClkBit) peripheral clock periods.

70.3.10.9 Arbitration and matching timing

During normal reception and transmission, the matching, arbitration, move-in, and move-out processes are executed during certain time windows inside the CAN frame, as shown in the following figures.

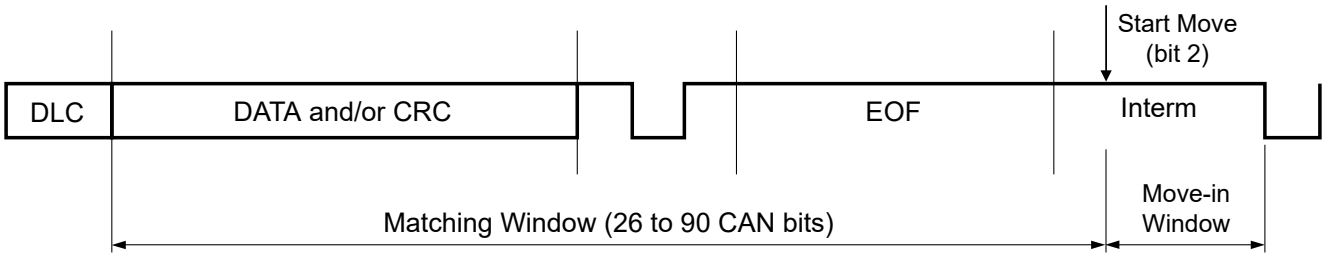


Figure 383. Matching and move-in time windows

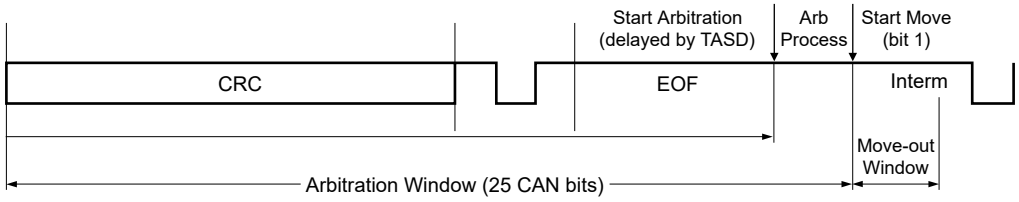


Figure 384. Arbitration and move-out time windows

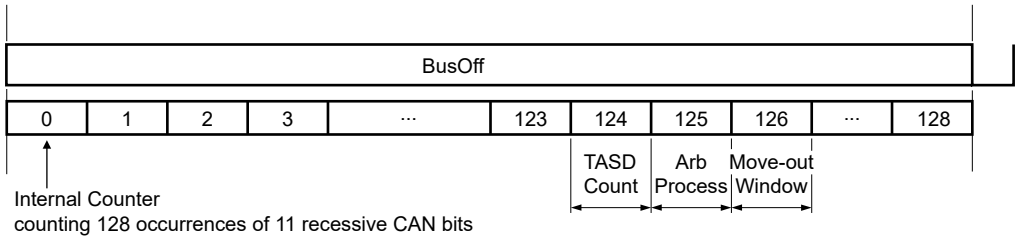


Figure 385. Arbitration at the end of bus off and move-out time windows

NOTE

In the preceding figures, the matching and arbitration timing does not take into account the delay caused by the concurrent memory access due to the CPU or other internal FlexCAN subblocks.

70.3.10.10 Tx arbitration start delay

CTRL2[TASD] (Tx Arbitration Start Delay) is a variable that indicates the number of CAN bits used by FlexCAN to delay the Tx arbitration process start point from the first bit of CRC field of the current frame. This variable can be written only in Freeze mode because it is blocked by hardware in other modes.

The transmission performance is impacted by the ability of the CPU to reconfigure message buffers (MBs) for transmission after the end of the internal arbitration process, where FlexCAN finds the winner MB for transmission (see Arbitration process). If the arbitration ends too early before the first bit of Intermission field, then there is a chance that the CPU reconfigures some Tx MBs and the winner MB is no longer the best candidate to be transmitted.

TASD is useful to optimize the transmission performance by defining the arbitration start point, as shown in the next figure, based on factors such as:

- Peripheral-to-oscillator clock ratio
- CAN bit timing variables that determine the CAN bit rate
- Number of message buffers (MBs) in use by the matching and arbitration processes

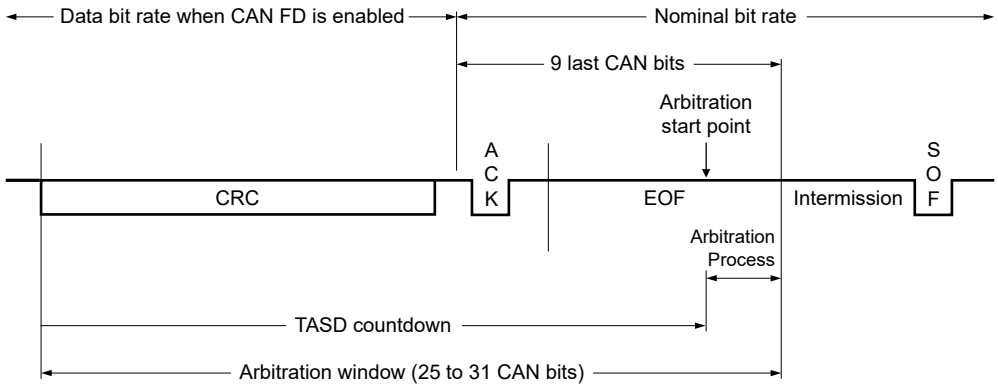


Figure 386. Optimal Tx Arbitration start point

The duration of an arbitration process, in terms of CAN bits, is directly proportional to the number of available MBs and to the CAN bit rate, and inversely proportional to the peripheral clock frequency.

The optimal arbitration timing is that in which the last MB is scanned just before the first bit of the Intermission field of a CAN frame. For instance, if there are few MBs and the peripheral/oscillator clock ratio is high and the CAN baud rate is low, then the arbitration can be placed closer to the frame's end, adding more delay to its start point, and vice-versa.

If TASD is set to 0 then the arbitration start is not delayed and more time is reserved for arbitration. On the other hand, if TASD is close to 24 then the CPU can configure a Tx MB later and less time is reserved for arbitration. If too little time is reserved for arbitration the FlexCAN may be not be able to find a winner MB in time to be transmitted with the best chance to win the bus arbitration against external nodes on the CAN bus.

The optimal TASD value can be calculated as follows:

For CAN FD frames and $(MAXMB + 1) \leq NMB_{END}$

$$TASD = 31 - \frac{2 * (MAXMB + 1) + 4}{CPCB_N}$$

For CAN FD frames and $(MAXMB + 1) > NMB_{END}$

$$TASD = 22 - \frac{2 * (MAXMB + 1) - NMB_{END}}{CPCB_F}$$

For non-FD frames

$$TASD = 25 - \frac{2 * (MAXMB + 1) + 4}{CPCB}$$

Equation 14. Optimal value for TASD

where:

$$NMB_{END} = \frac{(9 * CPCB_N) - 4}{2}$$

$$BITRATE_N = \left(\frac{f_{CANCLK}}{[1 + (EPSEG1 + 1) + (EPSEG2 + 1) + (EPROPSEG + 1)\ ] \times (EPRESDIV + 1)} \right)$$

$$BITRATE_F = \left(\frac{f_{CANCLK}}{[1 + (FPSEG1 + 1) + (FPSEG2 + 1) + FPROPSEG] \times (FPRESDIV + 1)} \right)$$

$$CPCB_N = \frac{f_{SYS}}{BITRATE_N}$$

$$CPCB_F = \frac{f_{SYS}}{BITRATE_F}$$

$$CPCB = CPCB_N$$

Equation 15. Variables used in TASD calculation

- MAXMB is the value in CTRL1[MAXMB].
- NMB_{END} is the number of message buffers that can be scanned by the arbitration process during the 9 last CAN bits at the end of a frame (see the figure above).
- $BITRATE_N$ is the CAN bit rate in bits per second calculated by the nominal CAN bit time variables.
- $BITRATE_F$ is the CAN bit rate in bits per second calculated by the data CAN bit time variables.
- $CPCB_N$ is the number of peripheral clocks per CAN bit in nominal bit rate for CAN FD frames.
- $CPCB_F$ is the number of peripheral clocks per CAN bit in data bit rate for CAN FD frames.
- CPCB is the number of peripheral clocks per CAN bit for non-FD frames.
- f_{CANCLK} is the oscillator clock, in Hz.
- f_{SYS} is the peripheral clock, in Hz.
- EPSEG1 is the value in CBT[EPSEG1] (CTRL1[PSEG1] can also be used).
- EPSEG2 is the value in CBT[EPSEG2] (CTRL1[PSEG2] can also be used).
- EPROPSEG is the value in CBT[EPROPSEG] (CTRL1[PROPSEG] can also be used).
- EPRESDIV is the value in CBT[EPRESDIV] (CTRL1[PRESDIV] can also be used).
- FPSEG1 is the value in FDCBT[FPSEG1].
- FPSEG2 is the value in FDCBT[FPSEG2].
- FPROPSEG is the value in FDCBT[FPROPSEG].
- FPRESDIV is the value in FDCBT[FPRESDIV].
- NTSEG1 is the value in ENCBT[NTSEG1].
- NTSEG2 is the value in ENCBT[NTSEG2].
- ENPRESDIV is the value in EPRS[ENPRESDIV].
- DTSEG1 is the value in EDCBT[DTSEG1].
- DTSEG2 is the value in EDCBT[DTSEG2].
- EDPRESDIV is the value in EPRS[EDPRESDIV].

If CTRL2[BTE] is set, then:

$$BITRATE_N = \frac{f_{CANCLK}}{[1 + (NTSEG1 + 1) + (NTSEG2 + 1)] \times (ENPRESDIV + 1)}$$

Equation 16. Nominal baud rate when CTRL2[BTE] = 1

$$BITRATE_F = \frac{f_{CANCLK}}{[1 + (DTSEG1 + 1) + (DTSEG2 + 1)] \times (EDPRESDIV + 1)}$$

Equation 17. Fast baud rate when CTRL2[BTE] = 1

See also Protocol timing for more details.

The following tables give the TASD value calculated for some configuration cases.

Case 1:

- Clock ratio = 2:1 (example: peripheral clock 80 MHz and oscillator clock 40 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 428. TASD values:

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	24	8.0
64	23	8.0
96	22	8.0

Case 2:

- Clock ratio = 1:1 (example: peripheral clock 40 MHz and oscillator clock 40 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 429. TASD values:

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	23	6.67
54	22	5.0
64	21	3.33
96	20	1.6

Case 3:

- Clock ratio = 2:1 (example: peripheral clock 40 MHz and oscillator clock 20 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 430. TASD values:

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	23	4.0
54	22	4.0
64	21	3.33
96	20	1.54

70.3.11 Clocks

The following table describes the clock sources for FlexCAN. Please see chip clocking chapter for clock setting, configuration and gating information.

Table 431. FlexCAN clocks

Clock name	Description
ipg_clk	Peripheral clock
ipg_clk_chi	Control Host Interface (CHI) clock
ipg_clk_pe	Protocol Engine (PE) clock
ipg_clk_pe_nogate	Protocol Engine clock (no gating)
ipg_clk_s	Peripheral access clock

70.3.11.1 Clock domains and restrictions

The FlexCAN module has two clock domains asynchronous to each other:

- The bus domain feeds the Control Host Interface (CHI) submodule.
- The oscillator domain feeds the CAN Protocol Engine (PE) submodule.

When the two domains are connected to clocks with different frequencies and/or phases, there are restrictions on the frequency relationship between the two clock domains. In the case of asynchronous operation, the bus domain clock frequency must always be greater than the oscillator domain clock frequency.

NOTE

Asynchronous operation with a 1:1 ratio between peripheral and oscillator clocks is not allowed.

When doing matching and arbitration, FlexCAN needs to scan the whole message buffer memory during the time slot of one CAN frame, comprised of a number of CAN bits. In order to have sufficient time to do that, the following requirements must be observed:

- The peripheral clock frequency cannot be smaller than the oscillator clock frequency.
- There must be a minimum number of peripheral clocks per CAN bit, as specified in the table shown below.

Table 432. Minimum number of peripheral clocks per CAN bit for Classical CAN format

Number of mailboxes	Value of MCR[RFEN]	Value of ERFCR[ERFEN]	Minimum number of peripheral clocks per CAN bit
16	0	0	16
32	0	0	16
64	0	0	25
96	0	0	37
16	1	0	16
32	1	0	17
64	1	0	30
96	1	0	42
16	0	1	16
32	0	1	19
64	0	1	31
96	0	1	42

For classical frame format, the minimum number of peripheral clocks per CAN bit specified in the preceding table determines the minimum peripheral clock frequency for a given number of mailboxes and for an expected CAN bit rate. The CAN bit rate depends on the number of time quanta in a CAN bit, which can be defined by adjusting one or more of the bit timing values contained in either the Control 1 register (CTRL1) or CAN Bit Time register (CBT) or Enhanced Nominal CAN bit timing register (ENCBT). The time quantum (Tq) is defined in Protocol timing. The minimum number of time quanta per CAN bit must be 8; therefore, the oscillator clock frequency should be at least 8 times the CAN bit rate.

For CAN FD frame format, there are some constraints that need to be satisfied. The number of peripheral clocks per CAN bit in nominal bit rate (NumClkNomBit) can be calculated by the equation below.

$$\text{NumClkNomBit} = \frac{f_{SYS}}{f_{CANCLK}} \times (\text{PRESDIV} + 1) \times (\text{PROPSEG} + \text{PSEG1} + \text{PSEG2} + 4)$$

$$= \frac{f_{SYS}}{\text{NomBitRate}}$$

Equation 18. Number of peripheral clocks per nominal CAN bit

where PRESDIV, PSEG1, and PSEG2 are CAN bit time values in CTRL1 register. Alternatively, EPRESDIV, EPSEG1, and EPSEG2 values in CBT register or the values of EPRS[ENPRESDIV], ENCBT[NTSEG1], and ENCBT[NTSEG2] can be used instead. NumClkNomBit can also be calculated as a function of the expected nominal bit rate used in the arbitration phase (NomBitRate), as shown in the equation above.

The number of CAN bits in the data phase of an FD frame with the BRS bit set (fast CAN bits, in short) depends on the number of data bytes in the payload. The number of fast CAN bits (NumOfFastBits) can be determined in the table below. The fewer the number of data bytes, the fewer the number of fast CAN bits, and less time is available for FlexCAN to scan the whole message buffer memory during the internal matching and arbitration processes.

Table 433. Number of fast CAN bits in a CAN FD frame

Minimum number of data bytes	DLC field	NumOfFastBits
0	0h	21
1	1h	29
2	2h	37
3	3h	45
4	4h	53
5	5h	61
6	6h	69
7	7h	77
8	8h	85
12	9h	117
16	Ah	149
20	Bh	186
24	Ch	218
32	Dh	282
48	Eh	410
64	Fh	538

The critical part of a CAN FD frame is during the data phase, where the CAN bit rate is faster than in the arbitration phase. The minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit) can be calculated to guarantee that enough time is available for FlexCAN to scan the message buffer memory during reception and transmission. The equation below calculates this constraint.

$$MinNumClkFastBit_A = \frac{(8.5 \times MaxNumOfMb) + [ERFEN \times (2 \times NFE + 4)] + 64 - (9 \times NumClkNomBit)}{NumOfFastBits}$$

Equation 19. Minimum number of peripheral clocks per fast CAN bit for FlexCAN scan process

where MaxNumOfMb is the maximum number of available mailboxes defined in MCR[MAXMB]. NFE and ERFEN are the fields defined in ERFCR register.

The clock domain crossing circuit between the CHI and PE subblocks also imposes a minimum number of peripheral clocks per fast CAN bit for the handshake mechanism to work properly without losing status information through the interface, as shown in the equation below.

$$MinNumClkFastBit_B = 3 \times \left(1 + \frac{f_{SYS}}{f_{CANCLK}}\right)$$

Equation 20. Minimum number of peripheral clocks per fast CAN bit for FlexCAN clock domain interface

Therefore, the minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit) is determined by the larger of the two values calculated above.

$$MinNumClkFastBit = Maximum\ (\ MinNumClkFastBit_A,\ MinNumClkFastBit_B\ )$$

Equation 21. Minimum number of peripheral clocks per fast CAN bit

Then, the maximum CAN bit rate in the data phase of CAN FD frames (DataBitRateMAX) can be calculated as below.

$$DataBitrate_{MAX} = \frac{f_{CANCLK}}{ROUNDUP\left(\frac{MinNumClkFastBit \times f_{CANCLK}}{f_{SYS}}\right)}$$

Equation 22. Maximum achievable baud rate for data phase

The peripheral and oscillator clock frequencies, the maximum number of mailboxes, and the expected nominal bit rate affect the maximum data bit rate attainable by FlexCAN in CAN FD mode. Also, the data bit rate depends on the minimum payload size of FD frames used in a given application.

To illustrate how the CAN FD bit rate is affected by the configuration of FlexCAN variables, an application example with the peripheral and oscillator clock frequencies set to 50 MHz and 40 MHz, respectively, is considered.

1. Considering the nominal bit rate as 1 Mbps, the number of peripheral clocks per CAN bit in nominal bit rate is calculated as below.

   $$NumClkNomBit = \frac{50 \times 10^6}{1 \times 10^6} = 50$$

   Equation 23. Calculation example for number of peripheral clocks per nominal CAN bit

2. The number of fast CAN bits (NumOfFastBits) is determined in the table presented above. For example, if the minimum payload in FD frames is 8 bytes, then there are 85 CAN bits in the data phase.
3. Assuming the maximum number of mailboxes is 96, and Enhanced Rx FIFO is disabled, the minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit) can be calculated.

   $$MinNumClkFastBit_A = \frac{(8.5 \times 96) + 64 - (9 \times 50)}{85} = 5.06$$

   Equation 24. Calculation example for number of peripheral clocks per fast CAN bit for FlexCAN scan process

   $$MinNumClkFastBit_B = 3 \times \left(1 + \frac{50}{40}\right) = 6.75$$

   Equation 25. Calculation example for number of peripheral clocks per fast CAN bit for FlexCAN clock domain interface

   $$MinNumClkFastBit = Maximum\ (5.06,\ 6.75) = 6.75$$

   Equation 26. Calculation example for number of peripheral clocks per fast CAN bit

4. The maximum CAN bit rate in the data phase can finally be found.

$$DataBitRate_{MAX} = \frac{40 \times 10^6}{ROUNDUP\left(\frac{6.75 \times 40 \times 10^6}{50 \times 10^6}\right)} = 6.667 \text{ Mbps}$$

Equation 27. Calculation example for maximum achievable baud rate

As demonstrated in this example, even though the oscillator clock frequency (40 MHz) is adequate to generate a data rate of 8 Mbps in CAN FD mode, the specific FlexCAN configuration limits this rate to 6.667 Mbps. This limitation is mainly due to the low peripheral clock frequency that imposes the MinNumClkFastBitB bound.

The table below shows the maximum data rate for CAN FD with Enhanced RX FIFO disabled according to clock frequencies, payload size, and number of available mailboxes. See in this table that, for some cases, if the number of available mailboxes is reduced, the FlexCAN can then achieve a data rate up to 8 Mbps.

Table 434. Maximum CAN bit rate in data phase on CAN FD frames with Enhanced Rx FIFO disabled

Peripheral clock frequency (MHz)	Payload size	Number of available mailboxes	Maximum data rate (Mbps)
40	8	94	6.667
40	8	114	5.0
40	12	117	6.667
40	12	128	5.714
50	12 to 64	128	6.667
60	8	126	8.0
60	12	128	8.0
67	6	128	8.0
80	3	128	8.0
100	0	128	8.0

70.3.12 Reset

You can reset FlexCAN in the following ways:

1. Chip level hard reset, which resets all memory-mapped registers asynchronously.
2. MCR[SOFTRST], which resets some of the memory-mapped registers synchronously. See Table 437 to see what registers are affected by soft reset.

Soft reset is synchronous and has to follow an internal request/acknowledge procedure across clock domains. Therefore, it may take some time to fully propagate its effects. MCR[SOFTRST] remains asserted while soft reset is pending, so software can poll this bit to know when the reset has completed. Also, soft reset cannot be applied while clocks are shut down in a low-power mode. The low-power mode should be exited and the clocks resumed before applying soft reset.

When the module is enabled (MCR[MDIS] negated), FlexCAN automatically enters Freeze mode. In Freeze mode, FlexCAN is un-synchronized to the CAN bus, MCR[HALT] and MCR[FRZ] are set, the internal state machines are disabled, then MCR[FRZACK] and MCR[NOTRDY] are set. The Tx pin is in recessive state and FlexCAN does not initiate any transmission or

reception of CAN frames. Note that the message buffers and the Rx Individual Mask registers are not affected by reset, so they are not automatically initialized.

70.3.13 Interrupts

The module has many interrupt sources: interrupts due to message buffers and interrupts due to the ORed interrupts from MBs, Bus Off, Bus Off Done, Error, Error Fast (errors detected in the data phase of CAN FD format messages with the BRS bit set), Tx Warning, and Rx Warning.

Each one of the message buffers can be an interrupt source, if its corresponding IMASK bit is set. There is no distinction between Tx and Rx interrupts for a particular buffer, under the assumption that the buffer is initialized for either transmission or reception. Each of the buffers has an assigned flag bit in the IFLAG registers. The bit is set when the corresponding buffer completes a successful transfer and is cleared when the CPU writes one to it (unless another interrupt is generated at the same time).

NOTE

It must be guaranteed that the CPU clears only the bit causing the current interrupt. For this reason, bit manipulation instructions (BSET) must not be used to clear interrupt flags. These instructions may cause accidental clearing of interrupt flags which are set after entering the current interrupt service routine.

If the Legacy Rx FIFO is enabled (MCR[RFEN] = 1) and DMA is disabled (MCR[DMA] = 0), the interrupts corresponding to MBs 0 to 7 have different meanings. Bit 7 of the IFLAG1 register becomes the "Legacy FIFO Overflow" flag; bit 6 becomes the "Legacy FIFO Warning" flag, bit 5 becomes the "Frames Available in Legacy FIFO" flag and bits 4-0 are unused. See the description of Interrupt Flags 1 Register (IFLAG1) for more information.

If both Legacy Rx FIFO and DMA are enabled (MCR[RFEN] and MCR[DMA] = 1) the FlexCAN does not generate any Legacy FIFO interrupt. Bit 5 of the IFLAG1 register still indicates "Frames Available in Legacy FIFO" and generates a DMA request. Bits 7, 6, 4-0 are unused.

CAUTION

Legacy FIFO cannot be enabled when CAN FD feature is enabled.

For a combined interrupt where multiple MB interrupt sources are OR'd together, the interrupt is generated when any of the associated MBs (or FIFO, if applicable) generates an interrupt. In this case, the CPU must read the IFLAG registers to determine which MB or FIFO source caused the interrupt.

The following interrupt sources generate interrupts like the MB interrupt sources, and can be read from ESR1 register

- Bus Off
- Bus Off Done
- Error
- Error Fast
- Tx Warning
- Rx Warning

The Bus Off, Error, Tx Warning, and Rx Warning interrupt mask bits are located in the CTRL1 register.

70.3.14 Bus interface

CPU access to FlexCAN registers is subject to the following rules:

- Unrestricted read and write access to supervisor registers (registers identified with S/U in Table 437 in Supervisor Mode or with S only) results in an access error.
- Read and write access to implemented reserved address space results in an access error.
- Write access to positions whose bits are all currently read-only results in an access error. If at least one of the bits is not read-only then no access error is issued. Write permission to positions or some of their bits can change depending on the mode of operation or transitory state. See register and field descriptions for details.

- Read and write access to unimplemented address space results in access error.
- Read and write access to RAM located positions during Low Power mode results in an access error.
- It is possible for the RXIMR memory region to be considered as general purpose memory and available for access. There are two ways of doing this:
  1. If MCR[IRMQ] is cleared, the individual masks (RXIMR) are disabled. In this case the RXIMR memory region is considered as general purpose memory.
  2. If MCR[MAXMB] is programmed with a value smaller than the available number of MBs, then the unused memory space can be used as general purpose RAM space. Note that reserved words within RAM cannot be used. As an example, suppose FlexCAN's RAM can support up to 16 MBs, CTRL2[RFFN] is 0h, and MCR[MAXMB] is programmed with zero. The maximum number of MBs in this case becomes one. The RAM starts at 0080h, and the space 0080h–008Fh is used by the one MB. The memory space 0090h–017Fh is available. The space 0180h–087Fh is reserved. The space 0880h–0883h is used by the one individual mask and the available memory in the mask registers space would be 0884h–08BFh. From 08C0h–09DFh there are reserved words for internal use which cannot be used as general purpose RAM. As a general rule, free memory space for general purpose depends only on MAXMB.
  3. If MCR[FDEN] is enabled, general purpose memory can be used out of Freeze mode only.

Table 435. Access permissions

Supervisor access	0				1		
Supervisor mode - MCR[SUPV] = 1	0			1	any		
Modes of operation	Normal	Freeze	Low power	*	Normal	Freeze	Low power
MCR	bus error	bus error	bus error	bus error	read/write	read/write	read/write
CTRL1	read/write	read write	read/write	bus error	read/write	read/write	read/write
TIMER	read/write	read/write	read/write	bus error	read/write	read/write	read/write
TCR	bus error	bus error	bus error	bus error	bus error	bus error	bus error
RXMGMASK[1]	bus error for write operation	read/write	bus error for write operation	bus error	bus error for write operation	read/write	bus error for write operation
RX14MASK[1]	bus error for write operation	read/write	bus error for write operation	bus error	bus error for write operation	read/write	bus error for write operation
RX15MASK[1]	bus error for write operation	read/write	bus error for write operation	bus error	bus error for write operation	read/write	bus error for write operation
ECR	bus error for write operation	read/write	bus error for write operation	bus error	bus error for write operation	read/write	bus error for write operation
ESR1	read/write	read/write	read/write	bus error	read/write	read/write	read/write
IMASK2	read/write	read/write	read/write	bus error	read/write	read/write	read/write

Table continues on the next page...

Table 435. Access permissions (continued)

Supervisor access	0			1			
Supervisor mode - MCR[SUPV] = 1	0			1	any		
Modes of operation	Normal	Freeze	Low power	*	Normal	Freeze	Low power
IMASK1	read/write	read/write	read/write	bus error	read/write	read/write	read/write
IFLAG2	read/write	read/write	read/write	bus error	read/write	read/write	read/write
IFLAG1	read/write	read/write	read/write	bus error	read/write	read/write	read/write
CTRL2	read/write	read/write	read/write	bus error	read/write	read/write	read/write
ESR2	read/write	read/write	read/write	bus error	read/write	read/write	read/write
CRCR	bus error for write operation	bus error for write operation	bus error for write operation	bus error	bus error for write operation	bus error for write operation	bus error for write operation
RXFGMASK[1]	bus error for write operation	read/write	bus error for write operation	bus error	bus error for write operation	read/write	bus error for write operation
RXFIR[1]	bus error for write operation	bus error for write operation	bus error for write operation	bus error	bus error for write operation	bus error for write operation	bus error for write operation
CBT	bus error for write operation	read/write	bus error for write operation	bus error	bus error for write operation	read/write	bus error for write operation
DBG1	bus error for write operation	bus error for write operation	bus error for write operation	bus error for write operation	bus error for write operation	bus error for write operation	bus error for write operation
DBG2	bus error for write operation	bus error for write operation	bus error for write operation	bus error for write operation	bus error for write operation	bus error for write operation	bus error for write operation
IMASK3	read/write	read/write	read/write	bus error	read/write	read/write	read/write
IFLAG3	read/write	read/write	read/write	bus error	read/write	read/write	read/write
MB[1] [2]	read/write	read/write	read/write	bus error	read/write	read/write	read/write
Legacy FIFO header[1]	read/write	read/write	read/write	bus error	read/write	read/write	read/write
Legacy FIFO reserved space[1]	bus error	bus error	bus error	bus error	bus error	bus error	bus error
Legacy FIFO filters[1]	read/write	read/write	read/write	bus error	read/write	read/write	read/write
RXIMR[1]	bus error	read/write	bus error	bus error	bus error	read/write	bus error

Table continues on the next page...

Table 435. Access permissions (continued)

Supervisor access	0				1		
Supervisor mode - MCR[SUPV] = 1	0			1	any		
Modes of operation	Normal	Freeze	Low power	*	Normal	Freeze	Low power
MECR	read/write	read/write	read/write	bus error	read/write	read/write	read/write
ERRIAR	read/write	read/write	read/write	bus error	read/write	read/write	read/write
ERRIDPR	read/write	read/write	read/write	bus error	read/write	read/write	read/write
ERRIPPR	read/write	read/write	read/write	bus error	read/write	read/write	read/write
RERRAR [3]	read/write	read/write	read/write	bus error	read/write	read/write	read/write
RERRDR[3]	read/write	read/write	read/write	bus error	read/write	read/write	read/write
RERRSYNR[3]	read/write	read/write	read/write	bus error	read/write	read/write	read/write
ERRSR	read/write	read/write	read/write	bus error	read/write	read/write	read/write
FDCTRL	read/write	read/write	read/write	bus error	read/write	read/write	read/write
FDCBT	read/write[3]	read/write	read/write[3]	bus error	read/write[3]	read/write	read/write[3]
FDCRC	bus error for write operation	bus error for write operation	bus error for write operation	bus error	bus error for write operation	bus error for write operation	bus error for write operation
ERFCR	read/write[3]	read/write	read/write[3]	bus error	read/write[3]	read/write	read/write[3]
ERFIER	read/write	read/write	read/write	bus error	read/write	read/write	read/write
ERFSR	read/write	read/write	read/write	bus error	read/write	read/write	read/write
HR_TIME_STAMP	read/write	read/write	read/write	bus error	read/write	read/write	read/write
Enhanced Rx FIFO header[4]	bus error for write operation	bus error for write operation	bus error for write operation	bus error	bus error for write operation	bus error for write operation	bus error for write operation
Enhanced Rx FIFO reserved space[4]	bus error	bus error	bus error	bus error	bus error	bus error	bus error
ERFFEL	bus error for write operation	read/write	bus error for write operation	bus error	bus error for write operation	read/write	bus error for write operation
General purpose RAM[1]	read/write	read/write	read/write	bus error	read/write	read/write	read/write
Reserved space (used)[1]	bus error	bus error	bus error	bus error	bus error	bus error	bus error
Reserved space (empty)[1]	bus error	bus error	bus error	bus error	bus error	bus error	bus error

1. Access in low power is only possible if RAM_clk and ipg_clk are enabled.
2. If MCR[RFEN] = 1, see Legacy FIFO access rules below.

3. Write operation has no effect.
4. Enhanced Rx FIFO must not be accessed if MCR[RFEN] is set.

70.3.15 Detection and correction of memory errors

NOTE

All FlexCAN memory must be initialized before starting its operation in order to have the parity bits in memory properly updated. CTRL2[WRMFRZ] grants write access to all memory positions that require initialization, ranging from 080h–ADFh and from C20h–31FFh. The RXMGMASK, RX14MASK, RX15MASK, and RXFGMASK registers need to be initialized as well. MCR[RFEN] and ERF[ERFEN] must not be set during memory initialization.

FlexCAN supports detection and correction of errors in memory read accesses. Each byte of FlexCAN memory is associated to five parity bits that ensure a Hamming distance of four. The error correction mechanism ensures that in this 13-bit word, errors in one bit can be corrected (correctable errors) and errors in two bits can be detected but not corrected (non-correctable errors). Errors in more than two bits may not be detected. In case of non-correctable errors, the corrupted data is not changed by the error correction logic. When a read access is performed, the parity bits are used to calculate a syndrome, which indicates the error in each byte.

FlexCAN detects a noncorrectable error in the event either an all-zeros or an all-ones read occurs. See description of Error Report Syndrome Register (RERRSYNR).

Memory errors are indicated to the host through status register (Error Status Register (ERRSR)) and bus transfer errors, and reported through report registers (Error Report Address register (RERRAR), Error Report Data register (RERRDR) and Error Report Syndrome register (RERRSYNR)).

The error detection and correction mechanism can be activated or not, controlled by MECR[ECCDIS]. When disabled, updates on indications and reporting registers are stopped, but the parity bits are still calculated and written along with data in memory write operations to ensure that memory has consistent parity bits associated with the data.

To avoid accidentally changing the critical error correction configuration, this protocol must be followed to enable the update of the Memory Error Control register (MECR):

1. By default, CTRL2[ECRWRE] is zero and MECR[ECRWRDIS] is one.
2. Set CTRL2[ECRWRE].
3. Clear MECR[ECRWRDIS].
4. All writes to Memory Error Control register (MECR) must keep MECR[ECRWRDIS] cleared.
5. After configuration is done, lock the Memory Error Control register (MECR) by either setting MECR[ECRWRDIS] or clearing CTRL2[ECRWRE].

70.3.15.1 Sources of the memory access

The FlexCAN memory can be accessed by two major sources (or requestors):

- Host (CPU): the largest word accessed is 32-bit.
- FlexCAN internal processes (Rx matching, Tx arbitration, move-in on reception, move-out on transmission): the largest word accessed is 64-bit.

The way that noncorrectable errors are indicated and reported depends on the source of access.

70.3.15.2 Error indication

Memory errors are indicated by flags HANCEIF, FANCEIF, and CEIF in the Error Status Register (ERRSR). Noncorrectable errors detected in memory reads requested by the host are indicated separately than the ones detected in requests by FlexCAN internal processes. FlexCAN makes no distinction of the source of the access when correctable errors are detected. There are three independent flags for these three cases, and each flag will raise an interrupt unless it is masked by mask bits in Memory Error Control Register (MECR). If both noncorrectable and correctable errors are found in different bytes in the same read operation, both flags are set.

A noncorrectable error detected in host access is also indicated as a bus transfer error. A bus wait request may be asserted to extend the memory transaction to the moment the report registers are updated. This indication cannot be masked. If the flag bit ERRSR[HANCEIF] is not masked, the same noncorrectable error will raise a bus transfer error and an interrupt request.

Each indication flag has one overrun flag in Error Status Register (ERRSR). The overrun flags do not request interrupts. Overrun flags for noncorrectable errors indicate that other errors of the same nature were detected after current error being treated; overrun flags for correctable errors indicate that other errors of the same nature were detected before the current error being treated.

This is the recommended handling sequence for error indication:

1. Get error report information from report registers.
2. Use this information to take proper measures in the application.
3. Clear the HANCEIF, FANCEIF, and CEIF flags.
4. If the overrun flag is active:
    a. Alert application that at least one error could not be managed.
    b. Clear the overrun flag.

The FlexCAN internal processes can access memory in transactions larger than 32 bits. For the indication, this kind of access is considered a consecutive sequence of 32-bit accesses. If errors are found in two or more 32-bit words the interrupt and overrun flags are set simultaneously.

70.3.15.3 Error reporting

The report registers Error Report Address register (RERRAR), Error Report Data register (RERRDR), and Error Report Syndrome register (RERRSYNR) provide detailed information about the address read, raw data, and syndrome read with error, and are indicated by the flags described in Error indication. The address, data, and syndrome registers are updated simultaneously along with the error flags, according to these rules:

1. If either of the two non-correctable error flags is currently set, the report registers are not updated (the previous non-correctable error reporting is preserved).
2. Otherwise (either no error flag is currently set or only the correctable error flag is currently set), the report registers are updated according to the new error; or according to the most severe of new errors if non-correctable and correctable errors are simultaneously detected.

Reporting of errors detected in accesses larger than 32-bit follows the rules described in Error indication and in the description of Error Report Address Register (RERRAR).

The address reported in RERRAR and defined in ERRIAR are not the same listed in the module memory map. The relation between the reported addresses and the respective ones in the module memory map is shown in the description of Error Injection Address Register (ERRIAR).

Addresses reported when reading memory portions organized as FIFOs, such as the Legacy Rx FIFO Structure, Enhanced RX FIFO Structure and the Rx FIFO Information Register (RXFIR), refer to the address of the specific entry accessed in the FIFO, not to the FIFO base address.

To assure coherence of the error report registers it is necessary to turn off the report update by setting MECR[RERRDIS] before reading the report registers.

70.3.15.4 Response to errors

Correctable errors have no consequence on FlexCAN operation because affected data is corrected before its use by the host or FlexCAN internal processes.

For host-initiated reads, a noncorrectable error may affect the host, but does not affect FlexCAN operation.

Non-correctable errors detected on memory reads requested by the FlexCAN internal processes may result in incorrect operation depending on the state of MECR[NCEFAFRZ], as follows:

- During reception (either matching or move-in processes), when a noncorrectable error occurs, an incorrect destination may be selected to store the incoming frame, a corrupted frame may be stored in the correct destination, or both. If MECR[NCEFAFRZ] is set, FlexCAN stops operation automatically and enters in Freeze mode to prevent corrupted data from being treated as valid by FlexCAN internal processes. When MECR[NCEFAFRZ] is negated, FlexCAN continues working and a corrupted frame is received.
- During arbitration process, when a noncorrectable error occurs, either a non-highest priority Tx message buffer may be mistakenly selected for transmission or its data may be corrupted. If MECR[NCEFAFRZ] is set, FlexCAN stops operation automatically and enters Freeze mode before starting the move-out. When MECR[NCEFAFRZ] is negated, FlexCAN proceeds to move-out with a corrupted frame that will be transmitted on the CAN bus.
- During move-out process, when a noncorrectable error occurs, a corrupted frame is copied from the selected Tx MB that won the arbitration to the Tx SMB for transmission. If MECR[NCEFAFRZ] is set, FlexCAN stops operation automatically and enters Freeze mode before starting the transmission. When MECR[NCEFAFRZ] is negated, the corrupted frame is transferred from the Tx SMB to the Protocol Engine (PE) sub-block and is transmitted on the CAN bus.
- A noncorrectable error can also be detected beyond the move-out process, when Tx data is read from Tx SMB (buffer located in RAM) to be transferred to the PE subblock for transmission. In this case, a frame with corrupted ID and/or data is transmitted on the CAN bus.

  To prevent the frame from being successfully received by the external nodes, FlexCAN inverts all bits in the CRC field (CRC sequence plus CRC delimiter), and transmits an error flag just after CRC delimiter as a result of self-detecting a Bit1 error and a form error due to the CRC field inversion. When MECR[NCEFAFRZ] is set, FlexCAN stops operation automatically and enters Freeze mode just after the error frame. When MECR[NCEFAFRZ] is negated, FlexCAN may attempt to re-transmit the same frame, as long as no other higher priority Tx MB is subsequently configured for transmission.

  In the event the noncorrectable error persists, FlexCAN eventually reaches the Bus Off state because of consecutive error detections. ECR[TXERRCNT] is updated every time FlexCAN inverts the CRC field, causing errors as described above.

When MECR[NCEFAFRZ] is set and FlexCAN enters Freeze mode, only the CPU can cause FlexCAN to exit Freeze mode and resume Normal mode. The assertion of MECR[NCEFAFRZ] is the only way to prevent corrupted frames from being transmitted on the CAN bus up to the move-out internal process.

The error report registers can provide information to the application for customized management of these situations.

70.3.15.5 Error injection

The error injection registers ERRIAR, ERRIDPR, and ERRIPPR are used to inject errors in memory reads in order to force errors and consequently update the indication and reporting registers. The relation between the error injection addresses and the respective ones in the module memory map is shown in the description of Error Injection Address Register (ERRIAR).

The injection is done by flipping the data and parity bits correspondent to the bits in 1 in ERRIDPR and ERRIPPR. Injection can be selected specifically for memory accesses requested by the host or by FlexCAN internal processes.

In case of accesses larger than 32-bits, the EXTERRIE bit in Memory Error Control register (MECR) extends the injection pattern, replicating it in 32-bit words to fill the width of the access.

NOTE

It is very unlikely, but error injection may correct a bit with error. This will not raise the error flags and reports as expected.

To ensure coherence among error injection registers and avoid spurious error injections, MECR[HAERRIE] and MECR[FAERRIE] must be cleared when configuring the memory injection registers.

70.4 FlexCAN signal descriptions

The FlexCAN module has two I/O signals connected to the external chip pins. These signals are summarized in the following table and described in more detail in the next subsections.

Table 436. FlexCAN signal descriptions

Signal	Description	I/O
CAN Rx	CAN receive pin	Input
CAN Tx	CAN transmit pin	Output

70.4.1 CAN Rx

This pin is the receive pin from the CAN bus transceiver. Dominant state is represented by logic level 0. Recessive state is represented by logic level 1.

70.4.2 CAN Tx

This pin is the transmit pin to the CAN bus transceiver. Dominant state is represented by logic level 0. Recessive state is represented by logic level 1.

70.5 Initialization/application information

This section provides instructions for initializing the FlexCAN module.

70.5.1 FlexCAN initialization sequence

For any configuration change/initialization it is required that FlexCAN be put into Freeze mode (see Freeze mode). Note that the module needs to be initialized after every reset.

The following is a generic initialization sequence applicable to the FlexCAN module:

1. Initialize the Module Configuration register (MCR).
    a. Enable the individual filtering per MB and reception queue features by setting IRMQ.
    b. Enable the warning interrupts by setting WRNEN.
    c. If required, disable frame self reception by setting SRXDIS.
    d. Enable the Legacy Rx FIFO by setting RFEN or the Enhanced Rx FIFO by setting ERFEN.
    e. If Legacy Rx FIFO or Enhanced Rx FIFO is enabled and DMA is required, set DMA.
    f. Enable the abort mechanism by setting AEN.
    g. Enable the local priority feature by setting LPRIOEN.
2. Initialize the Control 1 register (CTRL1) and optionally the CAN Bit Timing register (CBT). Initialize also the CAN FD CAN Bit Timing register (FDCBT).
    a. Determine the bit timing parameters: PROPSEG, PSEG1, PSEG2, and RJW.
    b. Optionally determine the bit timing parameters: EPROPSEG, EPSEG1, EPSEG2, and ERJW.
    c. Determine the CAN FD bit timing parameters: FPROPSEG, FPSEG1, FPSEG2, and FRJW.
    d. Determine the bit rate by programming the PRESDIV field and optionally the EPRESDIV field.
    e. Determine the CAN FD bit rate by programming the FPRESDIV field.
    f. Determine the internal arbitration mode (LBUF).
3. All FlexCAN memory must be initialized if Error Code Correction (ECC) is enabled. See Detection and correction of memory errors.
4. Initialize the message buffers.
    a. The control and status word of all message buffers must be initialized.

b. If Rx FIFO was enabled, the ID filter table must be initialized.

c. Other entries in each message buffer should be initialized as required.

5. Initialize the Rx Individual Mask registers (RXIMRn).
6. Set required interrupt mask bits in
   - IMASK registers (for all MB interrupts)
   - CTRL1 / CTRL2 registers (for Bus Off and Error interrupts)
7. Negate MCR[HALT].

After the last step listed above, FlexCAN attempts to synchronize to the CAN bus.

70.6 Memory map/register definition

This section describes the registers and data structures in the FlexCAN module. The base address of the module depends on the particular memory map of the chip.

70.6.1 FlexCAN memory mapping

The address space occupied by FlexCAN has 128 bytes for registers starting at the module base address, followed by embedded RAM starting at address offset 0080h.

Each individual register is identified by its complete name and the corresponding mnemonic . The access type can be Supervisor (S) or Unrestricted (U). Most of the registers can be configured to have either Supervisor or Unrestricted access by programming MCR[SUPV]. These registers are identified as S/U in the Access column of Table 437.

NOTE

An invalid register access will result in a bus error. This includes reading a write-only register, writing a read-only register, or accessing an invalid address.

Table 437. Register access and reset information

Register	Access type	Affected by hard reset	Affected by soft reset
Module Configuration Register (MCR)	S	Yes	Yes
Control 1 Register (CTRL1)	S/U	Yes	No
Free Running Timer Register (TIMER)	S/U	Yes	Yes
Rx Mailboxes Global Mask Register (RXMGMASK)	S/U	No	No
Rx Buffer 14 Mask Register (RX14MASK)	S/U	No	No
Rx Buffer 15 Mask Register (RX15MASK)	S/U	No	No
Error Counter Register (ECR)	S/U	Yes	Yes
Error and Status 1 Register (ESR1)	S/U	Yes	Yes
Interrupt Masks 2 Register (IMASK2)	S/U	Yes	Yes
Interrupt Masks 1 Register (IMASK1)	S/U	Yes	Yes

Table continues on the next page...

Table 437. Register access and reset information (continued)

Register	Access type	Affected by hard reset	Affected by soft reset
Interrupt Flags 2 Register (IFLAG2)	S/U	Yes	Yes
Interrupt Flags 1 Register (IFLAG1)	S/U	Yes	Yes
Control 2 Register (CTRL2)	S/U	Yes	No
Error and Status 2 Register (ESR2)	S/U	Yes	Yes
CRC Register (CRCR)	S/U	Yes	Yes
Rx FIFO Global Mask Register (RXFGMASK)	S/U	No	No
Rx FIFO Information Register (RXFIR)	S/U	No	No
CAN Bit Timing Register (CBT)	S/U	Yes	No
Interrupt Masks 3 Register (IMASK3)	S/U	Yes	Yes
Interrupt Flags 3 Register (IFLAG3)	S/U	Yes	Yes
Message buffers	S/U	No	No
Rx Individual Mask Registers	S/U	No	No
Memory Error Control Register (MECR)	S/U	Yes	Yes
Error Injection Address Register (ERRIAR)	S/U	Yes	Yes
Error Injection Data Pattern Register (ERRIDPR)	S/U	Yes	Yes
Error Injection Parity Pattern Register (ERRIPPR)	S/U	Yes	Yes
Error Report Address Register (RERRAR)	S/U	Yes	Yes
Error Report Data Register (RERRDR)	S/U	Yes	Yes
Error Report Syndrome Register (RERRSYNR)	S/U	Yes	Yes
Error Status Register (ERRSR)	S/U	Yes	Yes
Enhanced CAN Bit Timing Prescalers (EPRS)	S/U	Yes	No
Enhanced Nominal CAN Bit Timing (ENCBT)	S/U	Yes	No
Enhanced Data Phase CAN bit Timing (EDCBT)	S/U	Yes	No
Enhanced Transceiver Delay Compensation (ETDC)	S/U	Yes	No
CAN FD Control Register (FDCTRL)	S/U	Yes	No
CAN FD Bit Timing Register (FDCBT)	S/U	Yes	No

Table continues on the next page...

Table 437. Register access and reset information (continued)

Register	Access type	Affected by hard reset	Affected by soft reset
CAN FD CRC Register (FDCRC)	S/U	Yes	Yes
Enhanced Rx FIFO Control Register (ERFCR)	S/U	Yes	Yes
Enhanced Rx FIFO Interrupt Enable Register (ERFIER)	S/U	Yes	Yes
Enhanced Rx FIFO Status Register (ERFSR)	S/U	Yes	Yes
High Resolution Time Stamp (HR_TIME_STAMP)	S/U	No	No
Enhanced Rx FIFO	S/U	No	No
Enhanced Rx FIFO Filter Element (ERFFEL)	S/U	No	No

The FlexCAN module can store CAN messages for transmission and reception using mailboxes and Rx FIFO structures.

70.6.2 CAN register descriptions

The table below shows the FlexCAN memory map.

The address range from offset 80h–67Fh allocates the ninety-six 128-bit message buffers (MBs).

The memory maps for the message buffers are in FlexCAN message buffer memory map.

The address range from offset 2000h–204Ch allocates the Enhanced Rx FIFO output, and the address range from offset 2050h–263Ch allocates the rest of Enhanced RX FIFO 19 elements.

The memory map for the Enhanced Rx FIFO is in Enhanced Rx FIFO structure.

70.6.2.1 CAN memory map

CAN_0 base address: 4030_4000h

CAN_1 base address: 4030_8000h

CAN_2 base address: 4030_C000h

CAN_3 base address: 4031_0000h

CAN_4 base address: 4031_4000h

CAN_5 base address: 4031_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration Register (MCR)	32	RW	D890_000Fh
4h	Control 1 Register (CTRL1)	32	RW	0000_0000h
8h	Free Running Timer (TIMER)	32	RW	0000_0000h
10h	Rx Mailboxes Global Mask Register (RXMGMASK)	32	RW	See section
14h	Rx 14 Mask Register (RX14MASK)	32	RW	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
18h	Rx 15 Mask Register (RX15MASK)	32	RW	See section
1Ch	Error Counter (ECR)	32	RW	0000_0000h
20h	Error and Status 1 Register (ESR1)	32	RW	0000_0000h
24h	Interrupt Masks 2 Register (IMASK2)	32	RW	0000_0000h
28h	Interrupt Masks 1 Register (IMASK1)	32	RW	0000_0000h
2Ch	Interrupt Flags 2 Register (IFLAG2)	32	RW	0000_0000h
30h	Interrupt Flags 1 Register (IFLAG1)	32	RW	0000_0000h
34h	Control 2 Register (CTRL2)	32	RW	See section
38h	Error and Status 2 Register (ESR2)	32	R	0000_0000h
44h	CRC Register (CRCR)	32	R	0000_0000h
48h	Legacy Rx FIFO Global Mask Register (RXFGMASK)	32	RW	See section
4Ch	Legacy Rx FIFO Information Register (RXFIR)	32	R	See section
50h	CAN Bit Timing Register (CBT)	32	RW	0000_0000h
6Ch	Interrupt Masks 3 Register (IMASK3)	32	RW	0000_0000h
74h	Interrupt Flags 3 Register (IFLAG3)	32	RW	0000_0000h
880h - 9FCh	Rx Individual Mask Registers (RXIMR0 - RXIMR95)	32	RW	See section
AE0h	Memory Error Control Register (MECR)	32	RW	800C_0080h
AE4h	Error Injection Address Register (ERRIAR)	32	RW	0000_0000h
AE8h	Error Injection Data Pattern Register (ERRIDPR)	32	RW	0000_0000h
AECh	Error Injection Parity Pattern Register (ERRIPPR)	32	RW	0000_0000h
AF0h	Error Report Address Register (RERRAR)	32	R	0000_0000h
AF4h	Error Report Data Register (RERRDR)	32	R	0000_0000h
AF8h	Error Report Syndrome Register (RERRSYNR)	32	R	0000_0000h
AFCh	Error Status Register (ERRSR)	32	RW	0000_0000h
BF0h	Enhanced CAN Bit Timing Prescalers (EPRS)	32	RW	0000_0000h
BF4h	Enhanced Nominal CAN Bit Timing (ENCBT)	32	RW	0000_0000h
BF8h	Enhanced Data Phase CAN bit Timing (EDCBT)	32	RW	0000_0000h
BFCh	Enhanced Transceiver Delay Compensation (ETDC)	32	RW	0000_0000h
C00h	CAN FD Control Register (FDCTRL)	32	RW	8000_0100h
C04h	CAN FD Bit Timing Register (FDCBT)	32	RW	0000_0000h
C08h	CAN FD CRC Register (FDCRC)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
C0Ch	Enhanced Rx FIFO Control Register (ERFCR)	32	RW	0000_0000h
C10h	Enhanced Rx FIFO Interrupt Enable Register (ERFIER)	32	RW	0000_0000h
C14h	Enhanced Rx FIFO Status Register (ERFSR)	32	RW	0000_0000h
C30h - DACh	High Resolution Time Stamp (HR_TIME_STAMP0 - HR_TIME_STAMP95)	32	RW	See section
3000h - 31FCh	Enhanced Rx FIFO Filter Element (ERFFEL0 - ERFFEL127)	32	RW	See section

70.6.2.2 Module Configuration Register (MCR)

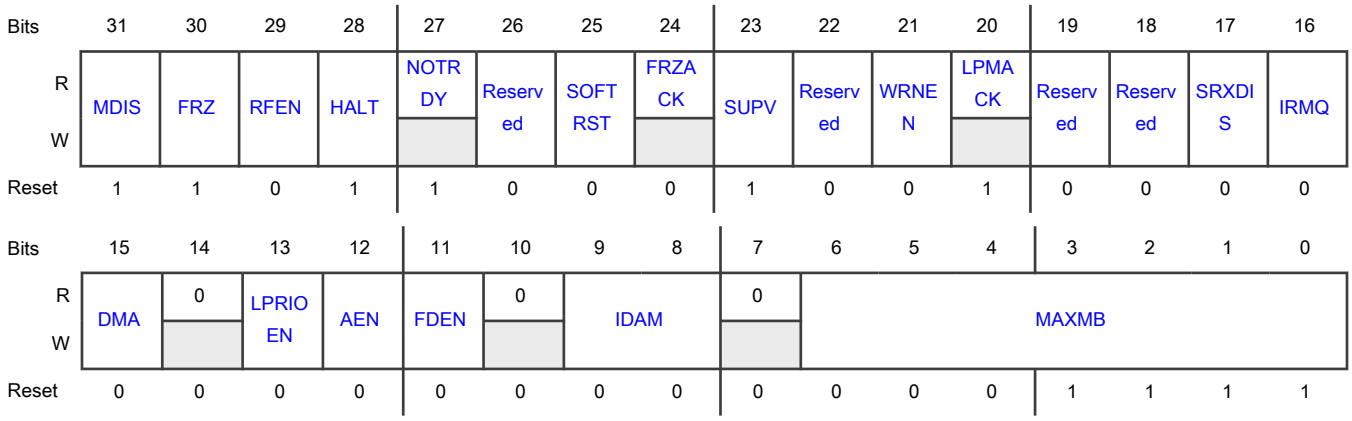
Offset

Register	Offset
MCR	0h

Function

This register defines global system configurations, such as the module operation modes and the maximum message buffer configuration.

Diagram



Fields

Field	Function
31 MDIS	Module Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit controls whether FlexCAN is enabled or not. When disabled, FlexCAN disables the clocks to the CAN Protocol Engine and Controller Host Interface submodules. This bit is not affected by soft reset. 0b - Enable the FlexCAN module. 1b - Disable the FlexCAN module.
30 FRZ	Freeze Enable The FRZ bit specifies the FlexCAN behavior when MCR[HALT] is set or when Debug mode is requested at chip level. When FRZ is asserted, FlexCAN is enabled to enter Freeze mode. Negation of this bit field causes FlexCAN to exit from Freeze mode. This bit is set by hardware when a non-correctable error is detected (MECR[NCEFAFRZ] is asserted). 0b - Not enabled to enter Freeze mode. 1b - Enabled to enter Freeze mode.
29 RFEN	Legacy Rx FIFO Enable This bit controls whether the Legacy Rx FIFO feature is enabled or not. When RFEN is set, MBs 0 to 5 cannot be used for normal reception and transmission because the corresponding memory region (80h–DCh) is used by the FIFO engine as well as additional MBs (up to 32, depending on CTRL2[RFFN] setting) which are used as Legacy Rx FIFO ID filter table elements. RFEN also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in Table 432. This bit can be written in Freeze mode only, because it is blocked by hardware in other modes. **NOTE** This bit cannot be set when CAN FD operation is enabled (see FDEN bit). **NOTE** RFEN bit must not be set if ERFCR[ERFEN] is set. **NOTE** The descriptions of the field settings vary by module instance.

Instance	Field value and description
CAN_0	0b - Legacy Rx FIFO not enabled. 1b - Legacy Rx FIFO enabled.
CAN_1	0b - Rx FIFO not enabled. 1b - Rx FIFO enabled.
CAN_2	0b - Rx FIFO not enabled. 1b - Rx FIFO enabled.
CAN_3	0b - Rx FIFO not enabled. 1b - Rx FIFO enabled.

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>CAN_4</td><td>0b - Rx FIFO not enabled. 1b - Rx FIFO enabled.</td></tr><tr><td>CAN_5</td><td>0b - Rx FIFO not enabled. 1b - Rx FIFO enabled.</td></tr></table>
28 HALT	Halt FlexCAN Assertion of this bit puts the FlexCAN module into Freeze mode. The CPU should clear it after initializing the message buffers and the Control registers CTRL1 and CTRL2. No reception or transmission is performed by FlexCAN before this bit is cleared. Freeze mode cannot be entered when FlexCAN is in a low power mode. The HALT bit is set by hardware when a non-correctable error is detected and MECR[NCEFAFRZ] is asserted. 0b - No Freeze mode request. 1b - Enters Freeze mode if the FRZ bit is asserted.
27 NOTRDY	FlexCAN Not Ready This read-only bit indicates that FlexCAN is either in Disable mode or Freeze mode. It is negated when FlexCAN has exited these modes. This bit is not affected by soft reset. 0b - FlexCAN module is either in Normal mode, Listen-Only mode, or Loop-Back mode. 1b - FlexCAN module is either in Disable mode or Freeze mode.
26 —	Reserved When writing to this field, always write the reset value.
25 SOFTRST	Soft Reset When SOFTRST is asserted, FlexCAN resets its internal state machines and some of the memory-mapped registers. SOFTRST can be asserted directly by the CPU when it writes to the MCR register. Because soft reset is synchronous and has to follow a request/acknowledge procedure across clock domains, it may take some time to fully propagate its effect. The SOFTRST bit remains asserted when reset is pending, and is automatically negated when reset completes. Therefore, software can poll this bit to know when the soft reset has completed. Soft reset cannot be applied when clocks are shut down in a low power mode. The module should be first removed from low power mode, and then soft reset can be applied. This bit is not affected by soft reset. 0b - No reset request. 1b - Resets the registers affected by soft reset.
24	Freeze Mode Acknowledge

Table continues on the next page...

Table continued from the previous page...

Field	Function
FRZACK	This read-only bit indicates that FlexCAN is in Freeze mode and its prescaler is stopped. The Freeze mode request cannot be granted until current transmission or reception processes have finished. Therefore the software can poll the FRZACK bit to know when FlexCAN has actually entered Freeze mode. If Freeze mode request is negated, then this bit is negated after the FlexCAN prescaler is running again. If Freeze mode is requested when FlexCAN is in a low power mode, then the FRZACK bit will be set only when the low-power mode is exited. See Freeze mode. This bit is not affected by soft reset. **NOTE** FRZACK will be asserted within 178 CAN bits from the Freeze mode request by the CPU, and negated within 2 CAN bits after the Freeze mode request removal (see Protocol timing). 0b - FlexCAN not in Freeze mode, prescaler running. 1b - FlexCAN in Freeze mode, prescaler stopped.
23 SUPV	Supervisor Mode This bit configures the FlexCAN to be either in Supervisor or User mode. The registers affected by this bit are marked as S/U in the "Access type" column of Table 437. Reset value of this bit is 1, so the affected registers start with Supervisor access allowance only. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0b - FlexCAN is in User mode. Affected registers allow both Supervisor and Unrestricted accesses. 1b - FlexCAN is in Supervisor mode. Affected registers allow only Supervisor access. Unrestricted access behaves as though the access was done to an unimplemented register location.
22 —	Reserved When writing to this field, always write the reset value.
21 WRNEN	Warning Interrupt Enable When asserted, this bit enables the generation of the TWRNINT and RWRNINT flags in the Error and Status Register 1 (ESR1). If WRNEN is negated, the TWRNINT and RWRNINT flags will always be zero, independent of the values of the error counters, and no warning interrupt will ever be generated. This bit can be written in Freeze mode only, because it is blocked by hardware in other modes. 0b - TWRNINT and RWRNINT bits are zero, independent of the values in the error counters. 1b - TWRNINT and RWRNINT bits are set when the respective error counter transitions from less than 96 to greater than or equal to 96.
20 LPMACK	Low-Power Mode Acknowledge This read-only bit indicates that FlexCAN is in a low-power mode (Disable mode). A low-power mode cannot be entered until all current transmission or reception processes have finished, so the CPU can poll the LPMACK bit to know when FlexCAN has actually entered low power mode. This bit is not affected by soft reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE LPMACK will be asserted within 180 CAN bits from the low-power mode request by the CPU, and negated within 2 CAN bits after the low-power mode request removal (see Protocol timing). 0b - FlexCAN is not in a low-power mode. 1b - FlexCAN is in a low-power mode.
19 —	Reserved When writing to this field, always write the reset value.
18 —	Reserved When writing to this field, always write the reset value.
17 SRXDIS	Self Reception Disable This bit defines whether FlexCAN is allowed to receive frames transmitted by itself. If this bit is asserted, frames transmitted by the module will not be stored in any MB, regardless if the MB is programmed with an ID that matches the transmitted frame, and no interrupt flag or interrupt signal will be generated due to the frame reception. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0b - Self-reception enabled. 1b - Self-reception disabled.
16 IRMQ	Individual Rx Masking And Queue Enable This bit indicates whether Rx matching process will be based either on individual masking and queue or on masking scheme with RXMGMASK, RX14MASK, RX15MASK, and RXFGMASK. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0b - Individual Rx masking and queue feature are disabled. For backward compatibility with legacy applications, the reading of C/S word locks the MB even if it is EMPTY. 1b - Individual Rx masking and queue feature are enabled.
15 DMA	DMA Enable DMA controls whether the DMA feature is enabled or not. The DMA feature can only be used in Legacy Rx FIFO or Enhanced Rx FIFO, so consequently MCR[RFEN] or ERFCR[ERFEN] must be asserted. When DMA and RFEN are set, IFLAG1[BUF5I] generates the DMA request and no RX FIFO interrupt is generated. This bit can be written in Freeze mode only as it is blocked by hardware in other modes. **NOTE** The descriptions of the field settings vary by module instance.

Instance	Field value and description
CAN_0	0b - DMA feature for Legacy RX FIFO or Enhanced Rx FIFO disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td></td><td>1b - DMA feature for Legacy RX FIFO or Enhanced Rx FIFO enabled.</td></tr><tr><td>CAN_1</td><td>0b - DMA feature for RX FIFO disabled. 1b - DMA feature for RX FIFO enabled.</td></tr><tr><td>CAN_2</td><td>0b - DMA feature for RX FIFO disabled. 1b - DMA feature for RX FIFO enabled.</td></tr><tr><td>CAN_3</td><td>0b - DMA feature for RX FIFO disabled. 1b - DMA feature for RX FIFO enabled.</td></tr><tr><td>CAN_4</td><td>0b - DMA feature for RX FIFO disabled. 1b - DMA feature for RX FIFO enabled.</td></tr><tr><td>CAN_5</td><td>0b - DMA feature for RX FIFO disabled. 1b - DMA feature for RX FIFO enabled.</td></tr></table>
14 —	Reserved
13 LPRIOEN	Local Priority Enable This bit is provided for backwards compatibility with legacy applications. It controls whether the local priority feature is enabled or not. It is used to expand the ID used during the arbitration process. With this expanded ID concept, the arbitration process is done based on the full 32-bit word, but the actual transmitted ID still has 11-bit for standard frames and 29-bit for extended frames. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0b - Local Priority disabled. 1b - Local Priority enabled.
12 AEN	Abort Enable When asserted, this bit enables the Tx abort mechanism. This mechanism guarantees a safe procedure for aborting a pending transmission, so that no frame is sent in the CAN bus without notification. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. **NOTE** When MCR[AEN] is asserted, only the abort mechanism (see Transmission abort mechanism) must be used for updating mailboxes configured for transmission.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	CAUTION Writing the Abort code into Rx mailboxes can cause unpredictable results when MCR[AEN] is asserted. 0b - Abort disabled. 1b - Abort enabled.
11 FDEN	CAN FD operation enable This bit enables the CAN with Flexible Data rate (CAN FD) operation. This bit can be written in Freeze mode only. **NOTE** FlexCAN is able to transmit FD frame format according to ISO 11898-1. **NOTE** The Legacy Rx FIFO Enable (RFEN) bit cannot be set if FDEN is asserted. 0b - CAN FD is disabled. FlexCAN is able to receive and transmit messages in CAN 2.0 format. 1b - CAN FD is enabled. FlexCAN is able to receive and transmit messages in both CAN FD and CAN 2.0 formats.
10 —	Reserved
9-8 IDAM	ID Acceptance Mode This 2-bit field identifies the format of the Legacy Rx FIFO ID filter table elements. Note that all elements of the table are configured at the same time by this field (they are all the same format). See Legacy Rx FIFO structure. This field can be written only in Freeze mode because it is blocked by hardware in other modes. 00b - Format A: One full ID (standard and extended) per ID filter table element. 01b - Format B: Two full standard IDs or two partial 14-bit (standard and extended) IDs per ID filter table element. 10b - Format C: Four partial 8-bit standard IDs per ID filter table element. 11b - Format D: All frames rejected.
7 —	Reserved
6-0 MAXMB	Number Of The Last Message Buffer This 7-bit field defines the number of the last message buffers that will take part in the matching and arbitration processes. The reset value (0Fh) is equivalent to a 16 MB configuration. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Number of the last MB = MAXMB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE MAXMB must be programmed with a value smaller than or equal to the number of available message buffers, as described in FlexCAN memory partition for CAN FD. Additionally, the definition of MAXMB value must take into account the region of MBs occupied by Legacy Rx FIFO and its ID filters table space defined by CTRL2[RFFN]. MAXMB also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in Table 432.

70.6.2.3 Control 1 Register (CTRL1)

Offset

Register	Offset
CTRL1	4h

Function

This Register is defined for specific FlexCAN control features related to the CAN bus, such as bit rate, programmable sampling point within an Rx bit, Loop Back mode, Listen-Only mode, Bus Off recovery behavior, and interrupt enabling (Bus-Off, Error, Warning). It also determines the division factor for the clock prescaler.

The CAN bit timing variables (PRESDIV, PROPSEG, PSEG1, PSEG2, and RJW) can also be configured in the CBT register, which extends the range of all these variables. If CBT[BTF] is set, PRESDIV, PROPSEG, PSEG1, PSEG2, and RJW fields of CTRL1 become read only.

If CTRL2[BTE] is set, then the PRESDIV, PROPSEG, PSEG1, PSEG2, and RJW fields in the CTRL1 register are not used by the hardware, are read as zero, and a write operation to them has no effect.

NOTE

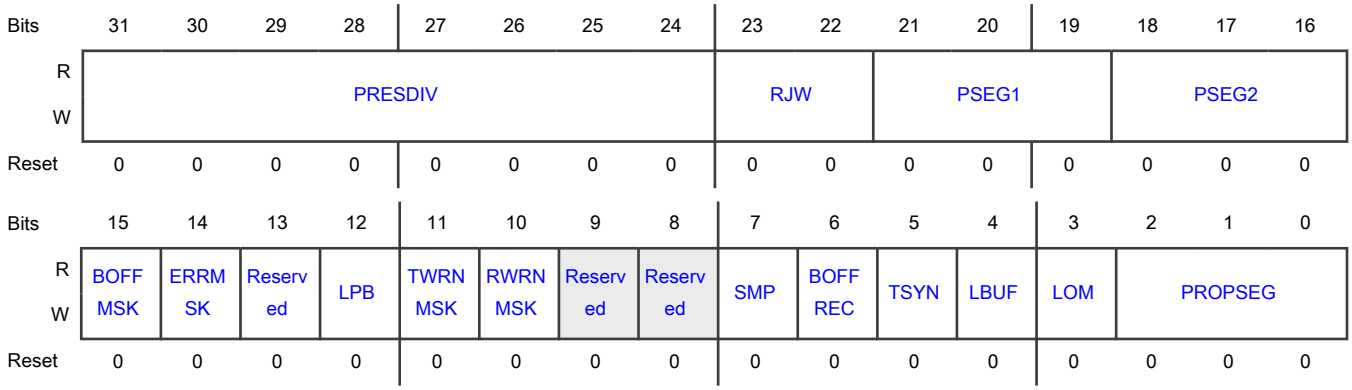
When the CAN FD feature is enabled, do not use the PRESDIV, RJW, PSEG1, PSEG2, and PROPSEG fields of the CTRL1 register for CAN bit timing. Instead use the CBT register's EPRESDIV, ERJW, EPSEG1, EPSEG2, and EPROPSEG fields.

NOTE

The CAN bit variables in CTRL1 and in CBT are stored in the same register.

The contents of this register are not affected by soft reset.

Diagram



Fields

Field	Function
31-24 PRESDIV	Prescaler Division Factor This 8-bit field defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency. The maximum value of this field is FFh, which gives a minimum Sclock frequency equal to the PE clock frequency divided by 256. See Protocol timing for more information. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Sclock frequency = PE clock frequency / (PRESDIV + 1).
23-22 RJW	Resync Jump Width This 2-bit field defines the maximum number of time quanta that a bit time can be changed by one resynchronization. One time quantum is equal to the Sclock period. The valid programmable values are 0–3. See Protocol timing for more information. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Resync Jump Width = RJW + 1.
21-19 PSEG1	Phase Segment 1 This 3-bit field defines the length of phase segment 1 in the bit time. The valid programmable values are 0–7. See Protocol timing for more information. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 1 = (PSEG1 + 1) × Time-Quanta.
18-16 PSEG2	Phase Segment 2 This 3-bit field defines the length of phase segment 2 in the bit time. The valid programmable values are 1–7. See Protocol timing for more information. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 2 = (PSEG2 + 1) × Time-Quanta.
15 BOFFMSK	Bus Off Interrupt Mask This bit provides a mask for the Bus Off interrupt ESR1[BOFFINT].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Bus Off interrupt disabled. 1b - Bus Off interrupt enabled.
14 ERRMSK	Error Interrupt Mask This bit provides a mask for the Error interrupt ESR1[ERRINT]. 0b - Error interrupt disabled. 1b - Error interrupt enabled.
13 —	Reserved **NOTE** This field is writeable only if the module is disabled. Otherwise the access type is read-only.
12 LPB	Loop Back Mode This bit configures FlexCAN to operate in Loop-Back mode. In this mode, FlexCAN performs an internal loop back that can be used for self-test operation. The bit stream output of the transmitter is fed back internally to the receiver input. The Rx CAN input pin is ignored and the Tx CAN output goes to the recessive state (logic 1). FlexCAN behaves as it normally does when transmitting, and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field, generating an internal acknowledge bit to ensure proper reception of its own message. Both transmit and receive interrupts are generated. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. **NOTE** In this mode, MCR[SRXDIS] cannot be asserted because this will impede the self-reception of a transmitted message. **NOTE** FDCTRL[TDCEN] and ETDC[ETDCEN] must be disabled when LPB is asserted. 0b - Loop Back disabled. 1b - Loop Back enabled.
11 TWRNMSK	Tx Warning Interrupt Mask This bit provides a mask for the Tx Warning interrupt associated with the TWRNINT flag in the Error and Status Register 1 (ESR1). This bit is read as zero when MCR[WRNEN] is negated. This bit can be written only if MCR[WRNEN] is asserted. 0b - Tx Warning interrupt disabled. 1b - Tx Warning interrupt enabled.
10 RWRNMSK	Rx Warning Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit provides a mask for the Rx Warning interrupt associated with the RWRNINT flag in the Error and Status Register 1 (ESR1). This bit is read as zero when MCR[WRNEN] bit is negated. This bit can be written only if MCR[WRNEN] bit is asserted. 0b - Rx Warning interrupt disabled. 1b - Rx Warning interrupt enabled.
9 —	Reserved
8 —	Reserved
7 SMP	CAN Bit Sampling This bit defines the sampling mode of CAN bits at the Rx input. It can be written in Freeze mode only, because it is blocked by hardware in other modes. **NOTE** For proper operation, to assert SMP it is necessary to guarantee a minimum value of two TQs in CTRL1[PSEG1] (or CBT[EPSEG1]).This bit cannot be asserted when CAN FD is enabled (MCR[FDEN] = 1). 0b - Just one sample is used to determine the bit value. 1b - Three samples are used to determine the value of the received bit: the regular one (sample point) and two preceding samples; a majority rule is used.
6 BOFFREC	Bus Off Recovery This bit defines how FlexCAN recovers from Bus Off state. If this bit is negated, automatic recovering from Bus Off state occurs according to the CAN Specification 2.0B. If the bit is asserted, automatic recovering from Bus Off is disabled and the module remains in Bus Off state until the bit is negated by the user. If the negation occurs before 128 sequences of 11 recessive bits are detected on the CAN bus, then Bus Off recovery happens as if the BOFFREC bit had never been asserted. If the negation occurs after 128 sequences of 11 recessive bits occurred, then FlexCAN will resynchronize to the bus by waiting for 11 recessive bits before joining the bus. After negation, the BOFFREC bit can be reasserted again during Bus Off, but it will be effective only the next time the module enters Bus Off. If BOFFREC was negated when the module entered Bus Off, asserting it during Bus Off will not be effective for the current Bus Off recovery. **NOTE** See Bus off in the CAN Protocol standard (ISO 11898-1) for details. 0b - Automatic recovering from Bus Off state enabled. 1b - Automatic recovering from Bus Off state disabled.
5 TSYN	Timer Sync

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit enables a mechanism that resets the free running timer each time a message is received in message buffer 0. This feature provides means to synchronize multiple FlexCAN stations with a special “SYNC” message, that is, global network time. If MCR[RFEN] is set (Legacy Rx FIFO enabled), the first available mailbox, according to CTRL2[RFFN] setting, is used for timer synchronization instead of MB0. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0b - Timer sync feature disabled 1b - Timer sync feature enabled
4 LBUF	Lowest Buffer Transmitted First This bit defines the ordering mechanism for message buffer transmission. When asserted, MCR[LPRIOEN] does not affect the priority arbitration. This bit can be written in Freeze mode only, because it is blocked by hardware in other modes. 0b - Buffer with highest priority is transmitted first. 1b - Lowest number buffer is transmitted first.
3 LOM	Listen-Only Mode This bit configures FlexCAN to operate in Listen-Only mode. In this mode, transmission is disabled, all error counters described in the ECR register are frozen, and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station will be received. If FlexCAN detects a message that has not been acknowledged, it will flag a BIT0 error without changing the receive error counter (ECR[RXERRCNT]), as if it was trying to acknowledge the message. Listen-Only mode is acknowledged by the state of ESR1[FLTCONF] indicating Passive Error. There can be some delay between the Listen-Only mode request and acknowledge. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0b - Listen-Only mode is deactivated. 1b - FlexCAN module operates in Listen-Only mode.
2-0 PROPSEG	Propagation Segment This 3-bit field defines the length of the propagation segment in the bit time. The valid programmable values are 0–7. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Propagation segment time = (PROPSEG + 1) × time-quanta. Time-quantum = one Sclock period.

70.6.2.4 Free Running Timer (TIMER)

Offset

Register	Offset
TIMER	8h

Function

This register represents a 16-bit free-running counter that can be read and written by the CPU. The timer starts from 0h after Reset, counts linearly to FFFFh, and wraps around.

When CTRL2[TIMER_SRC] is asserted, the timer is continuously incremented by an external time tick. The time tick must be synchronous to the peripheral clock, with a minimum pulse width of one clock cycle.

When CTRL2[TIMER_SRC] is negated, the timer is incremented by the CAN bit clock, which defines the baud rate on the CAN bus. During a message transmission/reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate. The timer is not incremented during Disable and Freeze modes.

The timer value is captured when the second bit of the identifier field of any frame is on the CAN bus. This captured value is written into the time stamp entry in a message buffer after a successful reception or transmission of a message.

If CTRL1[TSYN] is asserted, the timer is reset whenever a message is received in the first available mailbox, according to CTRL2[RFFN] setting.

The CPU can write to this register anytime. However, if the write occurs at the same time that the timer is being reset by a reception in the first mailbox, then the write value is discarded.

Reading this register affects the mailbox unlocking procedure (see Mailbox lock mechanism).

Diagram

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

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

Fields

Field	Function
31-16 —	Reserved
15-0 TIMER	Timer Value Contains the free-running counter value.

70.6.2.5 Rx Mailboxes Global Mask Register (RXMGMASK)

Offset

Register	Offset
RXMGMASK	10h

Function

This register is located in RAM.

RXMGMASK is provided for legacy application support.

- When MCR[IRMQ] is negated, RXMGMASK is always in effect (the bits in the MG field will mask the mailbox filter bits).
- When MCR[IRMQ] is asserted, RXMGMASK has no effect (the bits in the MG field will not mask the mailbox filter bits).

RXMGMASK is used to mask the filter fields of all Rx MBs, excluding MBs 14-15, which have individual mask registers.

This register can only be written in Freeze mode as it is blocked by hardware in other modes.

Diagram

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

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

Fields

Field	Function
31-0 MG	Rx Mailboxes Global Mask Bits These bits mask the mailbox filter bits. Note that the alignment with the ID word of the mailbox is not perfect as the two most significant MG bits affect the fields RTR and IDE, which are located in the Control and Status word of the mailbox. The following table shows in detail which MG bits mask each mailbox filter field.

SMB[RTR] [1]	CTRL2[RRS]	CTRL2[EACEN]	Mailbox filter fields			
			MB[RTR]	MB[IDE]	MB[ID]	Reserved
0	—	0	note [2]	note [3]	MG[28:0]	MG[31:29]
0	—	1	MG[31]	MG[30]	MG[28:0]	MG[29]
1	0	—	—	—	—	MG[31:0]
1	1	0	—	—	MG[28:0]	MG[31:29]
1	1	1	MG[31]	MG[30]	MG[28:0]	MG[29]

1. RTR bit of the incoming frame. It is saved into an auxiliary MB called Rx serial message buffer (Rx SMB).
2. If CTRL2[EACEN] is negated, the RTR bit of mailbox is never compared with the RTR bit of the incoming frame.
3. If CTRL2[EACEN] is negated, the IDE bit of mailbox is always compared with the IDE bit of the incoming frame.

Field	Function
	0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

70.6.2.6 Rx 14 Mask Register (RX14MASK)

Offset

Register	Offset
RX14MASK	14h

Function

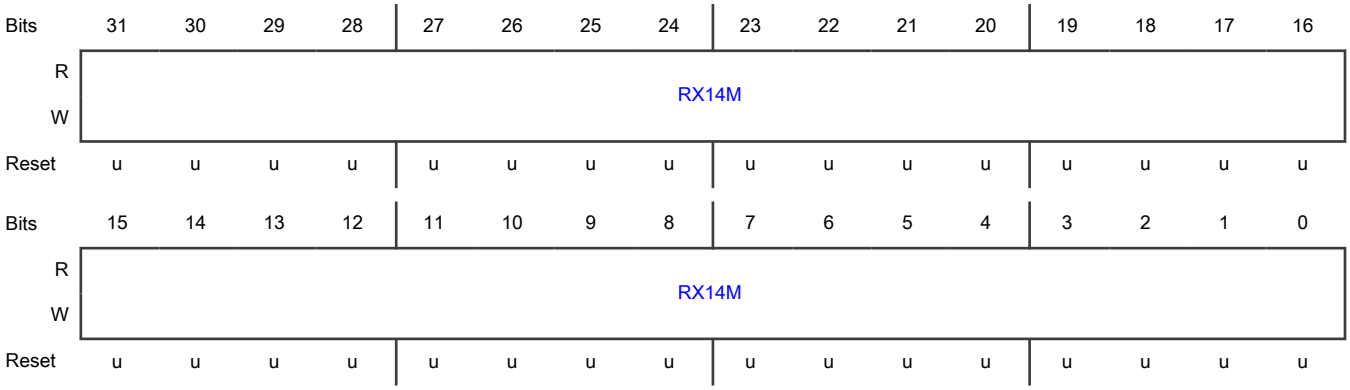
This register is located in RAM.

RX14MASK is provided for legacy application support. When MCR[IRMQ] is asserted, RX14MASK has no effect.

RX14MASK is used to mask the filter fields of message buffer 14.

This register can only be programmed when the module is in Freeze mode as it is blocked by hardware in other modes.

Diagram



Fields

Field	Function
31-0 RX14M	Rx Buffer 14 Mask Bits Each mask bit masks the corresponding mailbox 14 filter field in the same way that RXMGMASK masks other mailboxes' filters. See the description of the RXMGMASK register. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

70.6.2.7 Rx 15 Mask Register (RX15MASK)

Offset

Register	Offset
RX15MASK	18h

Function

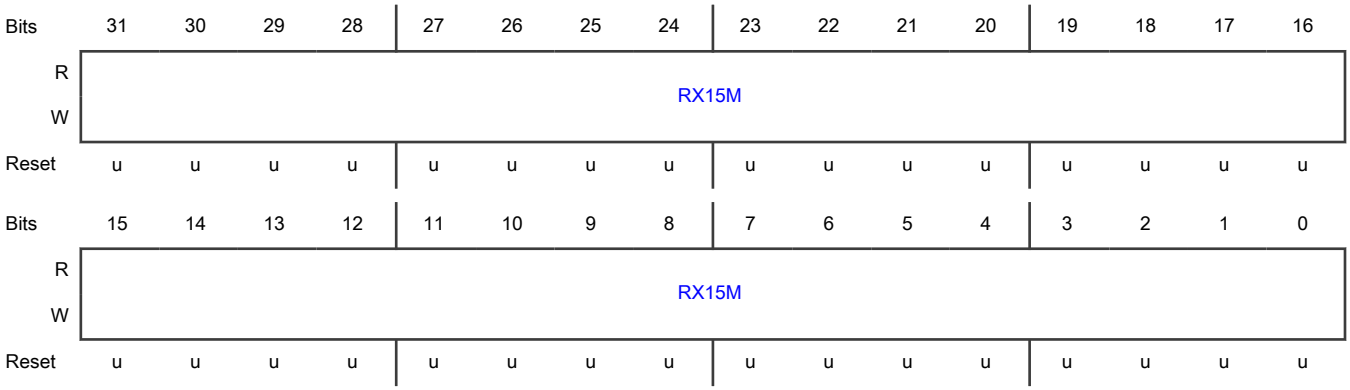
This register is located in RAM.

RX15MASK is provided for legacy application support. When MCR[IRMQ] is asserted, RX15MASK has no effect.

RX15MASK is used to mask the filter fields of message buffer 15.

This register can be programmed only when the module is in Freeze mode because it is blocked by hardware in other modes.

Diagram



Fields

Field	Function
31-0 RX15M	Rx Buffer 15 Mask Bits Each mask bit masks the corresponding mailbox 15 filter field in the same way that RXMGMASK masks other mailboxes' filters. See the description of the RXMGMASK register. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

70.6.2.8 Error Counter (ECR)

Offset

Register	Offset
ECR	1Ch

Function

This register has four 8-bit fields reflecting the value of the FlexCAN error counters:

- Transmit Error Counter (TXERRCNT field)
- Receive Error Counter (RXERRCNT field)
- Transmit Error Counter for errors detected in the data phase of CAN FD messages with the BRS bit set (TXERRCNT_FAST field)
- Receive Error Counter for errors detected in the data phase of CAN FD messages with the BRS bit set (RXERRCNT_FAST field)

The TXERRCNT and RXERRCNT counters take into account all errors in both CAN FD and non-FD message formats. TXERRCNT_FAST and RXERRCNT_FAST are dedicated to count only the errors that occur in the data phase of CAN FD frames with the BRS bit set.

The Fault Confinement state (ESR1[FLTCONF]) is updated based on TXERRCNT and RXERRCNT counters only. TXERRCNT and RXERRCNT counters can be written in Freeze mode only. TXERRCNT_FAST and RXERRCNT_FAST counters are read-only except in Freeze mode when the CPU can write value zero. The rules for increasing and decreasing these counters are described in the CAN protocol and are completely implemented in the FlexCAN module.

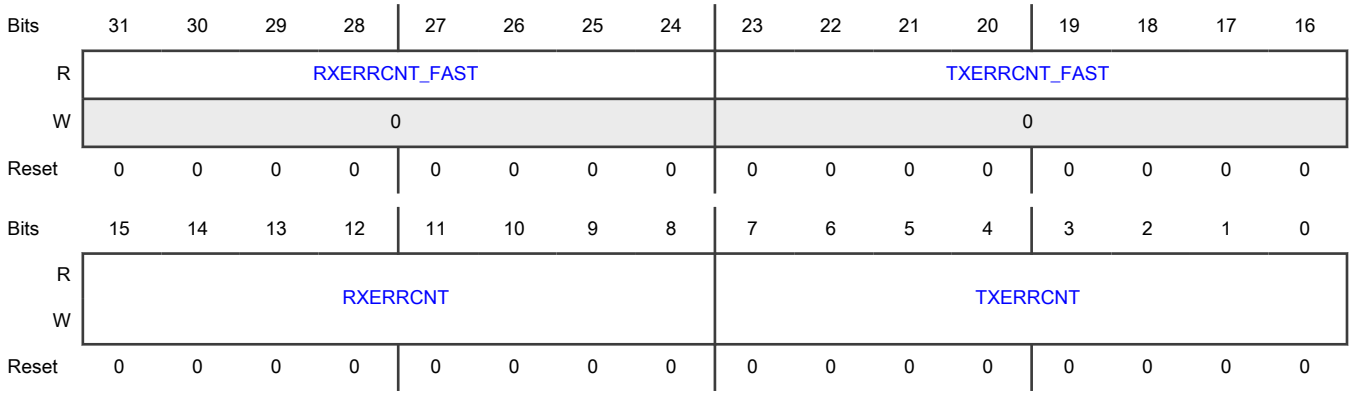
The following are the basic rules for FlexCAN bus state transitions:

- If the value of TXERRCNT or RXERRCNT becomes greater than or equal to 128, ESR1[FLTCONF] is updated to reflect Error Passive state.
- If the FlexCAN state is Error Passive, and either TXERRCNT or RXERRCNT decrements to a value less than or equal to 127 when the other already satisfies this condition, ESR1[FLTCONF] is updated to reflect Error Active state.
- If the value of TXERRCNT becomes greater than 255, ESR1[FLTCONF] is updated to reflect Bus Off state, and an interrupt may be issued. The value of TXERRCNT is then reset to zero.
- If FlexCAN is in Bus Off state, then TXERRCNT is cascaded together with another internal counter to count the 128th occurrences of 11 consecutive recessive bits on the bus. Hence, TXERRCNT is reset to zero and counts in a manner where the internal counter counts 11 such bits and then wraps around when incrementing the TXERRCNT. When TXERRCNT reaches the value of 128, ESR1[FLTCONF] is updated to be Error Active and both error counters are reset to zero. At any instance of dominant bit following a stream of less than 11 consecutive recessive bits, the internal counter resets itself to zero without affecting the TXERRCNT value. The TXERRCNT_FAST counter is frozen during Bus Off.
- If during system startup only one node is operating, then its TXERRCNT increases in each message it is trying to transmit, as a result of acknowledge errors (indicated by the ACKERR bit in the Error and Status Register). After the transition to Error Passive state, TXERRCNT does not increment anymore by acknowledge errors. Therefore the device never goes to the Bus Off state.
- If RXERRCNT increases to a value greater than 127, it is not incremented further, even if more errors are detected when being a receiver. At the next successful message reception, the counter is set to a value between 119 and 127 to return to Error Active state.
- TXERRCNT_FAST and RXERRCNT_FAST error counter values increment and decrement based on errors detected only in the data phase of CAN FD frames with the BRS bit set, following the same increment and decrement rules as TXERRCNT and RXERRCNT counters. These counters do not wrap around and get stuck at their maximum value (255). They stop counting and keep their values frozen when FlexCAN is in Bus Off state. They are reset when FlexCAN leaves Bus Off state and restart counting after FlexCAN returns to Error Active state.

NOTE

See Fault confinement in the CAN Protocol standard (ISO 11898-1) for details.

Diagram



Fields

Field	Function
31-24 RXERRCNT_FAST	Receive Error Counter for fast bits Receive error counter for errors detected in the data phase of received CAN FD messages with the BRS bit set. The RXERRCNT_FAST counter is read-only except in Freeze mode, where the CPU can write an 8-bit zero value only.
23-16 TXERRCNT_FAST	Transmit Error Counter for fast bits Transmit error counter for errors detected in the data phase of transmitted CAN FD messages with the BRS bit set. The TXERRCNT_FAST counter is read-only except in Freeze mode, where the CPU can write an 8-bit zero value only.
15-8 RXERRCNT	Receive Error Counter Receive error counter for all errors detected in received messages. The RXERRCNT counter is read-only except in Freeze mode, where it can be written by the CPU.
7-0 TXERRCNT	Transmit Error Counter Transmit error counter for all errors detected in transmitted messages. The TXERRCNT counter is read-only except in Freeze mode, where it can be written by the CPU.

70.6.2.9 Error and Status 1 Register (ESR1)

Offset

Register	Offset
ESR1	20h

Function

This register reports various error conditions detected in the reception and transmission of a CAN frame, some general status of the device, and is the source of some interrupts to the CPU. The reported error conditions are:

NOTE

Reading Host can clear these field.

- Errors detected in CAN frames of any format:
  - BIT1ERR
  - BIT0ERR
  - ACKKERR
  - CRCERR
  - FRMERR
  - STFERR
- Errors detected in the data phase of CAN FD frames with the BRS bit set only:
  - BIT1ERR_FAST
  - BIT0ERR_FAST
  - CRCERR_FAST
  - FRMERR_FAST
  - STFERR_FAST

An error detected in a single CAN frame may be reported by one or more error flags. Also, error reporting is cumulative, in case more error events happen in the next frames when the CPU does not attempt to read this register.

Status bits:

- TXWRN
- RXWRN
- IDLE
- TX
- FLTCONF
- RX
- SYNCH

Interrupt bits:

- BOFFINT
- BOFFDONEINT
- ERRINT
- ERRINT_FAST
- TWRNINT
- RWRNINT

It is recommended that the CPU use the following procedure when servicing interrupt requests generated by these bits:

1. Read this register to capture all error condition and status bits. This action clears the respective bits that were set since the last read access.
2. Write 1 to clear the interrupt bit that has triggered the interrupt request.
3. Write 1 to clear the ERR_OVR bit, if it is set.

Starting from all error flags cleared, a first error event sets either the ERRINT or the ERRINT_FAST (provided the corresponding mask bit is asserted). If other error events in subsequent frames happen before the CPU serves the interrupt request, the ERR_OVR bit is set to indicate that errors from different frames have accumulated.

Table 438. Can bus status

SYNCH	IDLE	TX	RX	FlexCAN state
0	0	0	0	Not synchronized to CAN bus
1	1	x	x	Idle
1	0	1	0	Transmitting
1	0	0	1	Receiving

NOTE

See Fault confinement in the CAN Protocol standard (ISO 11898-1) for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BIT1ERR...	BIT0ERR...	0	CRCERR_...	FRMERR_...	STFERR_...	0				ERROVR	ERRINT_...	BOFFDON...	SYNCH	TWRNINT	RWRNINT
W											W1C	W1C	W1C		W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BIT1ERR	BIT0ERR	ACKERR	CRCERR	FRMERR	STFERR	TXWRN	RXWRN	IDLE	TX	FLTCONF		RX	BOFFINT	ERRINT	0
W														W1C	W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BIT1ERR_FAST	Bit1 Error in the Data Phase of CAN FD frames with the BRS bit set This bit indicates when an inconsistency occurs between the transmitted and the received bit in the data phase of CAN FD frames with the BRS bit set. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit sent as recessive is received as dominant.
30 BIT0ERR_FAST	Bit0 Error in the Data Phase of CAN FD frames with the BRS bit set This bit indicates when an inconsistency occurs between the transmitted and the received bit in the data phase of CAN FD frames with the BRS bit set. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit sent as dominant is received as recessive.

Table continues on the next page...

Table continued from the previous page...

Field	Function
29 —	Reserved
28 CRCERR_FAST	Cyclic Redundancy Check Error in the CRC field of CAN FD frames with the BRS bit set This bit indicates that a CRC error has been detected by the receiver node in the CRC field of CAN FD frames with the BRS bit set, that is, the calculated CRC is different from the received. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A CRC error occurred since last read of this register.
27 FRMERR_FAST	Form Error in the Data Phase of CAN FD frames with the BRS bit set This bit indicates that a form error has been detected by the receiver node in the data phase of CAN FD frames with the BRS bit set, that is, a fixed-form bit field contains at least one illegal bit. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A form error occurred since last read of this register.
26 STFERR_FAST	Stuffing Error in the Data Phase of CAN FD frames with the BRS bit set This bit indicates that a stuffing error has been detected in the data phase of CAN FD frames with the BRS bit set. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A stuffing error occurred since last read of this register.
25-22 —	Reserved
21 ERROVR	Error Overrun This bit indicates that an error condition occurred when any error flag is already set. This bit is cleared by writing it to 1. 0b - Overrun has not occurred. 1b - Overrun has occurred.
20 ERRINT_FAST	Error interrupt for errors detected in Data Phase of CAN FD frames with BRS bit set This bit indicates that at least one of the error bits detected in the data phase of CAN FD frames with the BRS bit set (BIT1ERR_FAST, BIT0ERR_FAST, CRCERR_FAST, FRMERR_FAST, or STFERR_FAST) is set. If the corresponding mask bit CTRL2[ERRMSK_FAST] is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0b - No such occurrence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Indicates setting of any error bit detected in the data phase of CAN FD frames with the BRS bit set.
19 BOFFDONEINT	Bus Off Done Interrupt This bit is set when the Tx Error Counter (TXERRCNT) has finished counting 128 occurrences of 11 consecutive recessive bits on the CAN bus and is ready to leave Bus Off. If the corresponding mask bit in the Control 2 Register (BOFFDONEMSK) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0b - No such occurrence. 1b - FlexCAN module has completed Bus Off process.
18 SYNCH	CAN Synchronization Status This read-only flag indicates whether the FlexCAN is synchronized to the CAN bus and able to participate in the communication process. It is set and cleared by the FlexCAN. See the table in the overall ESR1 register description. 0b - FlexCAN is not synchronized to the CAN bus. 1b - FlexCAN is synchronized to the CAN bus.
17 TWRNINT	Tx Warning Interrupt Flag If MCR[WRNEN] is asserted, TWRNINT is set when the TXWRN flag transitions from 0 to 1, meaning that the Tx error counter reached 96. If the corresponding mask bit in the Control 1 Register (CTRL1[TWRNMSK]) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. When MCR[WRNEN] is negated, this flag is masked. CPU must clear this flag before disabling MCR[WRNEN]. Otherwise it will be set when MCR[WRNEN] is set again. Writing 0 has no effect. This flag is not generated during Bus Off state. This bit is not updated during Freeze mode. 0b - No such occurrence. 1b - The Tx error counter transitioned from less than 96 to greater than or equal to 96.
16 RWRNINT	Rx Warning Interrupt Flag If MCR[WRNEN] is asserted, RWRNINT is set when the RXWRN flag transitions from 0 to 1, meaning that the Rx error counters reached 96. If the corresponding mask bit in the Control 1 Register (CTRL1[RWRNMSK]) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. When WRNEN is negated, this flag is masked. CPU must clear this flag before disabling the bit. Otherwise it will be set when WRNEN is set again. Writing 0 has no effect. This bit is not updated during Freeze mode. 0b - No such occurrence. 1b - The Rx error counter transitioned from less than 96 to greater than or equal to 96.
15 BIT1ERR	Bit1 Error This bit indicates when an inconsistency occurs between the transmitted and the received bit in a non-CAN FD message or in the arbitration or data phase of a CAN FD message.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This bit is not set by a transmitter in case of arbitration field or ACK slot, or in case of a node sending a passive error flag that detects dominant bits. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit sent as recessive is received as dominant.
14 BIT0ERR	Bit0 Error This bit indicates when an inconsistency occurs between the transmitted and the received bit in a non-CAN FD message or in the arbitration or data phase of a CAN FD message. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit sent as dominant is received as recessive.
13 ACKERR	Acknowledge Error This bit indicates that an Acknowledge Error has been detected by the transmitter node, that is, a dominant bit has not been detected during the ACK SLOT. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - An ACK error occurred since last read of this register.
12 CRCERR	Cyclic Redundancy Check Error This bit indicates that a Cyclic Redundancy Check (CRC) error has been detected by the receiver node either in a non-FD message or in the arbitration or data phase of a frame in CAN FD format, that is, the calculated CRC is different from the received. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A CRC error occurred since last read of this register.
11 FRMERR	Form Error This bit indicates that a form error has been detected in a non-FD message or else in an FD message's arbitration or data phase by the receiver node, that is, a fixed-form bit field contains at least one illegal bit. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A Form Error occurred since last read of this register.
10 STFERR	Stuffing Error This bit indicates that a stuffing error has been detected in a non-FD message or else in an FD message's arbitration or data phase by the receiver node.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A stuffing error occurred since last read of this register.
9 TXWRN	TX Error Warning This bit indicates when repetitive errors are occurring during message transmission and is affected by the value of TXERRCNT in ECR register only. This bit is not updated during Freeze mode. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - TXERRCNT is greater than or equal to 96.
8 RXWRN	Rx Error Warning This bit indicates when repetitive errors are occurring during message reception and is affected by the value of RXERRCNT in ECR register only. This bit is not updated during Freeze mode. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - RXERRCNT is greater than or equal to 96.
7 IDLE	IDLE This bit indicates when CAN bus is in IDLE state. See the table in the overall ESR1 register description. 0b - No such occurrence. 1b - CAN bus is now IDLE.
6 TX	FlexCAN In Transmission This bit indicates if FlexCAN is transmitting a message. See the table in the overall ESR1 register description. 0b - FlexCAN is not transmitting a message. 1b - FlexCAN is transmitting a message.
5-4 FLTCONF	Fault Confinement State This 2-bit field indicates the confinement state of the FlexCAN module. If CTRL1[LOM] is asserted, after a delay that depends on the CAN bit timing, ESR1[FLTCONF] will indicate Error Passive. The very same delay affects the way that ESR1[FLTCONF] reflects an update to ECR register by the CPU. It may be necessary to wait up to one CAN bit time to get them coherent again. This bit field is affected by soft reset, but if the LOM bit is asserted, its reset value lasts just one CAN bit. After this time, ESR1[FLTCONF] reports Error Passive. 00b - Error Active 01b - Error Passive 1xb - Bus Off

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 RX	FlexCAN In Reception This bit indicates if FlexCAN is receiving a message. See the table in the overall ESR1 register description. 0b - FlexCAN is not receiving a message. 1b - FlexCAN is receiving a message.
2 BOFFINT	Bus Off Interrupt This bit is set when FlexCAN enters Bus Off state. If the corresponding mask bit in the Control Register 1 (CTRL1[BOFFMSK]) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0b - No such occurrence. 1b - FlexCAN module entered Bus Off state.
1 ERRINT	Error Interrupt This bit indicates that at least one of the error bits (BIT1ERR, BIT0ERR, ACKERR, CRCERR, FRMERR, or STFERR) is set. If the corresponding mask bit CTRL1[ERRMSK] is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0b - No such occurrence. 1b - Indicates setting of any error bit in the Error and Status register.
0 —	Reserved

70.6.2.10 Interrupt Masks 2 Register (IMASK2)

Offset

Register	Offset
IMASK2	24h

Function

This register allows any number of a range of the 32 message buffer interrupts to be enabled or disabled for MB63 to MB32. It contains one interrupt mask bit per buffer, enabling the CPU to determine which buffer generates an interrupt after a successful transmission or reception, that is, when the corresponding IFLAG2 bit is set.

NOTE

Each module instance supports a different number of registers.

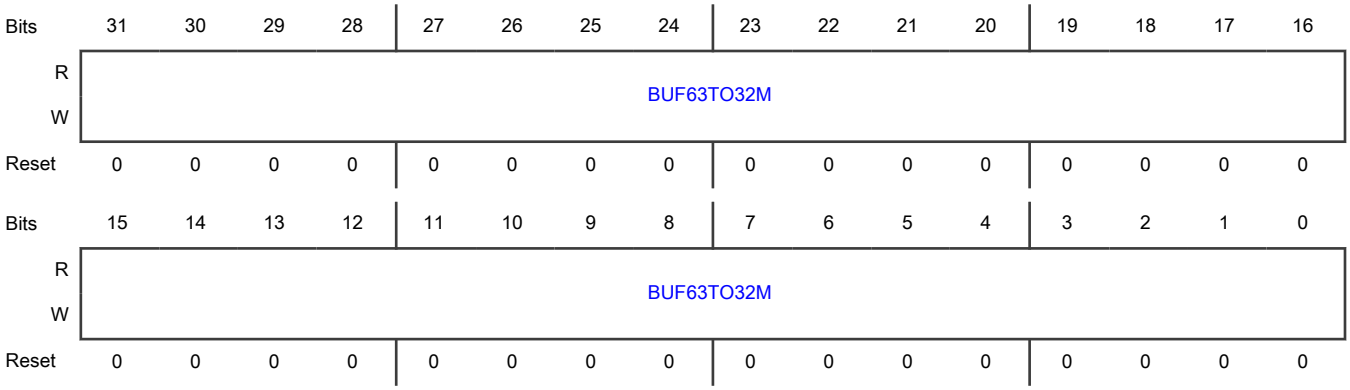
Instance	Register supported	Register not supported
CAN_0	IMASK2	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
CAN_1	IMASK2	—
CAN_2	IMASK2	—
CAN_3	—	IMASK2
CAN_4	—	IMASK2
CAN_5	—	IMASK2

Diagram



Fields

Field	Function
31-0 BUF63TO32M	Buffer MBi Mask Each bit enables or disables the corresponding FlexCAN message buffer interrupt for MB63 to MB32. **NOTE** Setting or clearing a bit in the IMASK2 Register can assert or negate an interrupt request, if the corresponding IFLAG2 bit is set. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

70.6.2.11 Interrupt Masks 1 Register (IMASK1)

Offset

Register	Offset
IMASK1	28h

Function

This register allows any number of a range of the 32 message buffer interrupts to be enabled or disabled for MB31 to MB0. It contains one interrupt mask bit per buffer, enabling the CPU to determine which buffer generates an interrupt after a successful transmission or reception, that is, when the corresponding IFLAG1 bit is set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BUF31TO0M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BUF31TO0M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BUF31TO0M	Buffer MBi Mask Each bit enables or disables the corresponding FlexCAN message buffer interrupt for MB31 to MB0. **NOTE** Setting or clearing a bit in the IMASK1 Register can assert or negate an interrupt request, if the corresponding IFLAG1 bit is set. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

70.6.2.12 Interrupt Flags 2 Register (IFLAG2)

Offset

Register	Offset
IFLAG2	2Ch

Function

This register defines the flags for the 32 message buffer interrupts for MB63 to MB32. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the respective IFLAG2 bit. If the corresponding IMASK2 bit is set, an interrupt will be generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect.

Before updating MCR[MAXMB], CPU must service the IFLAG2 bits whose MB value is greater than the MAXMB to be updated; otherwise, they will remain set and be inconsistent with the number of MBs available.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
CAN_0	IFLAG2	—
CAN_1	IFLAG2	—
CAN_2	IFLAG2	—
CAN_3	—	IFLAG2
CAN_4	—	IFLAG2
CAN_5	—	IFLAG2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BUF63TO32I															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BUF63TO32I															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BUF63TO32I	Buffer MBi Interrupt Each bit flags the corresponding FlexCAN message buffer interrupt for MB63 to MB32. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.

70.6.2.13 Interrupt Flags 1 Register (IFLAG1)

Offset

Register	Offset
IFLAG1	30h

Function

This register defines the flags for the 32 message buffer interrupts for MB31 to MB0. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding IFLAG1 bit. If the corresponding IMASK1 bit is set, an interrupt will be generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect. There is an exception when DMA for Legacy Rx FIFO is enabled, as described below.

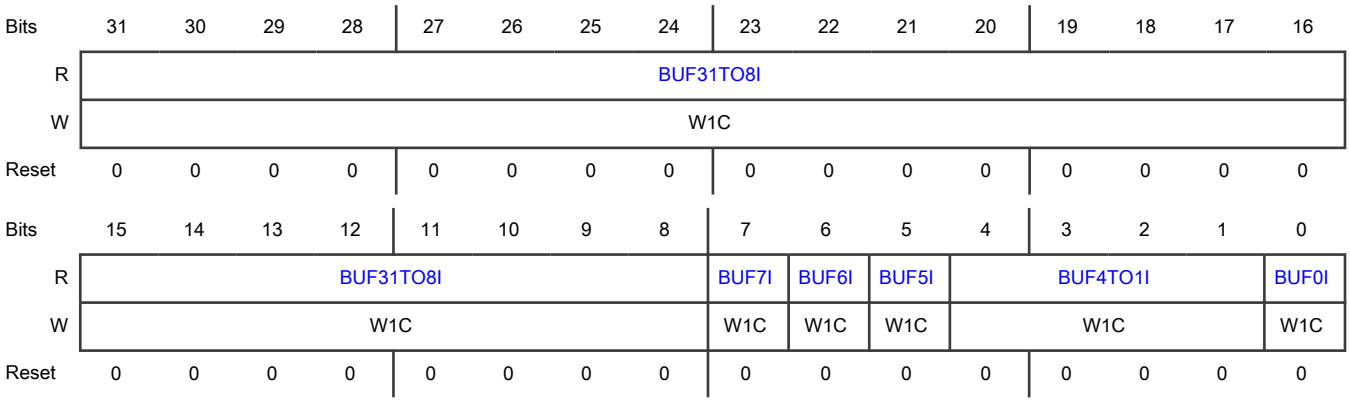
The BUF7I to BUF5I flags are also used to represent Legacy FIFO interrupts when the Legacy Rx FIFO is enabled. When MCR[RFEN] is set and MCR[DMA] is negated, the function of the eight least significant interrupt flags changes: BUF7I, BUF6I, and BUF5I indicate operating conditions of the Legacy FIFO , BUF0I is used to empty Legacy FIFO, and BUF4I to BUF1I bits are reserved.

Before enabling MCR[RFEN], the CPU must service the IFLAG bits asserted in the Legacy Rx FIFO region; see Legacy Rx FIFO. Otherwise, these IFLAG bits will mistakenly show the related MBs now belonging to Legacy FIFO as having contents to be serviced. When MCR[RFEN] is negated, the Legacy FIFO flags must be cleared. The same care must be taken when a CTRL2[RFFN] value is selected, extending Legacy Rx FIFO filters beyond MB7. For example, when RFFN is 8h, the MB0–23 range is occupied by Legacy Rx FIFO filters and related IFLAG bits must be cleared.

When both the MCR[RFEN] and MCR[DMA] bits are asserted (DMA feature for Legacy Rx FIFO enabled), the function of the eight least significant interrupt flags (BUF7I–BUF0I) are changed to support the DMA operation. BUF7I and BUF6I are not used, as well as BUF4I–BUF1I. BUF5I indicates operating condition of Legacy FIFO, and BUF0I is used to empty Legacy FIFO. Moreover, BUF5I does not generate a CPU interrupt, but generates a DMA request. IMASK1 bits in Legacy Rx FIFO region are not considered when bit MCR[DMA] is enabled. In addition the CPU must not clear the flag BUF5I when DMA is enabled. Before enabling MCR[DMA], the CPU must service the IFLAGs asserted in the Legacy Rx FIFO region. When MCR[DMA] is negated, the Legacy FIFO must be empty. Legacy FIFO must be disabled when MCR[FDEN] is enabled.

Before updating MCR[MAXMB], CPU must service the IFLAG1 bits whose MB value is greater than the MCR[MAXMB] to be updated; otherwise, they will remain set and be inconsistent with the number of MBs available.

Diagram



Fields

Field	Function
31-8 BUF31TO8I	Buffer MBi Interrupt Each bit flags the corresponding FlexCAN message buffer interrupt for MB31 to MB8. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 BUF7I	Buffer MB7 Interrupt Or Legacy Rx FIFO Overflow When MCR[RFEN] is cleared (Legacy Rx FIFO disabled), this bit flags the interrupt for MB7. **NOTE** This flag is cleared by the FlexCAN whenever MCR[RFEN] is changed by CPU writes. The BUF7I flag represents Legacy Rx FIFO overflow when MCR[RFEN] is set. In this case, the flag indicates that a message was lost because the Legacy Rx FIFO is full. Note that the flag will not be asserted when the Legacy Rx FIFO is full and the message was captured by a mailbox. **NOTE** The descriptions of the field settings vary by module instance.

Instance	Field value and description
CAN_0	0b - No occurrence of MB7 completing transmission/reception when MCR[RFEN]=0, or of Legacy Rx FIFO overflow when MCR[RFEN]=1 1b - MB7 completed transmission/reception when MCR[RFEN]=0, or Legacy Rx FIFO overflow when MCR[RFEN]=1
CAN_1	0b - No occurrence of MB7 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO overflow when MCR[RFEN]=1 1b - MB7 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO overflow when MCR[RFEN]=1
CAN_2	0b - No occurrence of MB7 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO overflow when MCR[RFEN]=1 1b - MB7 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO overflow when MCR[RFEN]=1
CAN_3	0b - No occurrence of MB7 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO overflow when MCR[RFEN]=1 1b - MB7 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO overflow when MCR[RFEN]=1
CAN_4	0b - No occurrence of MB7 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO overflow when MCR[RFEN]=1 1b - MB7 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO overflow when MCR[RFEN]=1
CAN_5	0b - No occurrence of MB7 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO overflow when MCR[RFEN]=1 1b - MB7 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO overflow when MCR[RFEN]=1

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 BUF6I	Buffer MB6 Interrupt Or Legacy Rx FIFO Warning When MCR[RFEN] is cleared (Legacy Rx FIFO disabled), this bit flags the interrupt for MB6. **NOTE** This flag is cleared by the FlexCAN whenever MCR[RFEN] is changed by CPU writes. The BUF6I flag represents Legacy Rx FIFO warning when MCR[RFEN] is set. In this case, the flag indicates when the number of unread messages within the Legacy Rx FIFO is increased to five from four due to the reception of a new one, meaning that the Legacy Rx FIFO is almost full. Note that if the flag is cleared when the number of unread messages is greater than four, it does not assert again until the number of unread messages within the Legacy Rx FIFO is decreased to be equal to or less than four. **NOTE** The descriptions of the field settings vary by module instance.

Instance	Field value and description
CAN_0	0b - No occurrence of MB6 completing transmission/reception when MCR[RFEN]=0, or of Legacy Rx FIFO almost full when MCR[RFEN]=1 1b - MB6 completed transmission/reception when MCR[RFEN]=0, or Legacy Rx FIFO almost full when MCR[RFEN]=1
CAN_1	0b - No occurrence of MB6 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO almost full when MCR[RFEN]=1 1b - MB6 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO almost full when MCR[RFEN]=1
CAN_2	0b - No occurrence of MB6 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO almost full when MCR[RFEN]=1 1b - MB6 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO almost full when MCR[RFEN]=1
CAN_3	0b - No occurrence of MB6 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO almost full when MCR[RFEN]=1 1b - MB6 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO almost full when MCR[RFEN]=1
CAN_4	0b - No occurrence of MB6 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO almost full when MCR[RFEN]=1 1b - MB6 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO almost full when MCR[RFEN]=1
CAN_5	0b - No occurrence of MB6 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO almost full when MCR[RFEN]=1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td></td><td>1b - MB6 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO almost full when MCR[RFEN]=1</td></tr></table>
5 BUF5I	Buffer MB5 Interrupt Or Frames available in Legacy Rx FIFO When MCR[RFEN] is cleared (Legacy Rx FIFO disabled), this field flags the interrupt for MB5. **NOTE** This flag is cleared by the FlexCAN whenever MCR[RFEN] is changed by CPU writes. When MCR[RFEN] is set (Legacy Rx FIFO enabled), the BUF5I flag represents "Frames available in Legacy Rx FIFO" and indicates that at least one frame is available to be read from the Legacy Rx FIFO. When the MCR[DMA] bit is enabled, this flag generates a DMA request and the CPU must not clear this bit by writing 1 in BUF5I. **NOTE** The descriptions of the field settings vary by module instance. <table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>CAN_0</td><td>0b - No occurrence of MB5 completing transmission/reception when MCR[RFEN]=0, or of frame(s) available in the Legacy FIFO, when MCR[RFEN]=1 1b - MB5 completed transmission/reception when MCR[RFEN]=0, or frame(s) available in the Legacy Rx FIFO when MCR[RFEN]=1. It generates a DMA request in case of MCR[RFEN] and MCR[DMA] are enabled.</td></tr><tr><td>CAN_1</td><td>0b - No occurrence of MB5 completing transmission/reception when MCR[RFEN]=0, or of frame(s) available in the FIFO, when MCR[RFEN]=1 1b - MB5 completed transmission/reception when MCR[RFEN]=0, or frame(s) available in the Rx FIFO when MCR[RFEN]=1. It generates a DMA request in case of MCR[RFEN] and MCR[DMA] are enabled.</td></tr><tr><td>CAN_2</td><td>0b - No occurrence of MB5 completing transmission/reception when MCR[RFEN]=0, or of frame(s) available in the FIFO, when MCR[RFEN]=1 1b - MB5 completed transmission/reception when MCR[RFEN]=0, or frame(s) available in the Rx FIFO when MCR[RFEN]=1. It generates a DMA request in case of MCR[RFEN] and MCR[DMA] are enabled.</td></tr></table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>CAN_3</td><td>0b - No occurrence of MB5 completing transmission/reception when MCR[RFEN]=0, or of frame(s) available in the FIFO, when MCR[RFEN]=1

1b - MB5 completed transmission/reception when MCR[RFEN]=0, or frame(s) available in the Rx FIFO when MCR[RFEN]=1. It generates a DMA request in case of MCR[RFEN] and MCR[DMA] are enabled.</td></tr>
<tr><td>CAN_4</td><td>0b - No occurrence of MB5 completing transmission/reception when MCR[RFEN]=0, or of frame(s) available in the FIFO, when MCR[RFEN]=1

1b - MB5 completed transmission/reception when MCR[RFEN]=0, or frame(s) available in the Rx FIFO when MCR[RFEN]=1. It generates a DMA request in case of MCR[RFEN] and MCR[DMA] are enabled.</td></tr>
<tr><td>CAN_5</td><td>0b - No occurrence of MB5 completing transmission/reception when MCR[RFEN]=0, or of frame(s) available in the FIFO, when MCR[RFEN]=1

1b - MB5 completed transmission/reception when MCR[RFEN]=0, or frame(s) available in the Rx FIFO when MCR[RFEN]=1. It generates a DMA request in case of MCR[RFEN] and MCR[DMA] are enabled.</td></tr>
</table>
</td></tr>
<tr><td>4-1
BUF4TO1I</td><td>Buffer MBi Interrupt Or Reserved

When MCR[RFEN] is cleared (Legacy Rx FIFO disabled), these bits flag the interrupts for MB4 to MB1.

NOTE
These flags are cleared by the FlexCAN whenever MCR[RFEN] is changed by CPU writes.

The BUF4TO1I flags are reserved when MCR[RFEN] is set.

0b - The corresponding buffer has no occurrence of successfully completed transmission or reception when MCR[RFEN]=0.

1b - The corresponding buffer has successfully completed transmission or reception when MCR[RFEN]=0.</td></tr>
<tr><td>0
BUF0I</td><td>Buffer MB0 Interrupt Or Clear Legacy FIFO bit

When MCR[RFEN] is cleared (Legacy Rx FIFO disabled), this field flags the interrupt for MB0. If the Legacy Rx FIFO is enabled, this bit is used to trigger the clear Legacy FIFO operation. This operation empties Legacy FIFO contents. Before performing this operation the CPU must service all Legacy FIFO related IFLAGs. When MCR[DMA] is enabled this operation also clears the BUF5I flag and consequently aborts the DMA request. The clear Legacy FIFO operation occurs when the CPU writes 1 in BUF0I. It is only allowed in Freeze mode and is blocked by hardware in other conditions.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - The corresponding buffer has no occurrence of successfully completed transmission or reception when MCR[RFEN]=0. 1b - The corresponding buffer has successfully completed transmission or reception when MCR[RFEN]=0.

70.6.2.14 Control 2 Register (CTRL2)

Offset

Register	Offset
CTRL2	34h

Function

This register complements Control1 register, providing control bits for memory write access in Freeze mode, for extending Legacy FIFO filter quantity, and to adjust the operation of internal FlexCAN processes like matching and arbitration.

The contents of this register are not affected by soft reset.

This table shows how the Legacy Rx FIFO filter structure is determined by the value of CTRL2[RFFN]. See the CTRL2[RFFN] field description for more information.

Table 439. Legacy Rx FIFO filter: possible structures

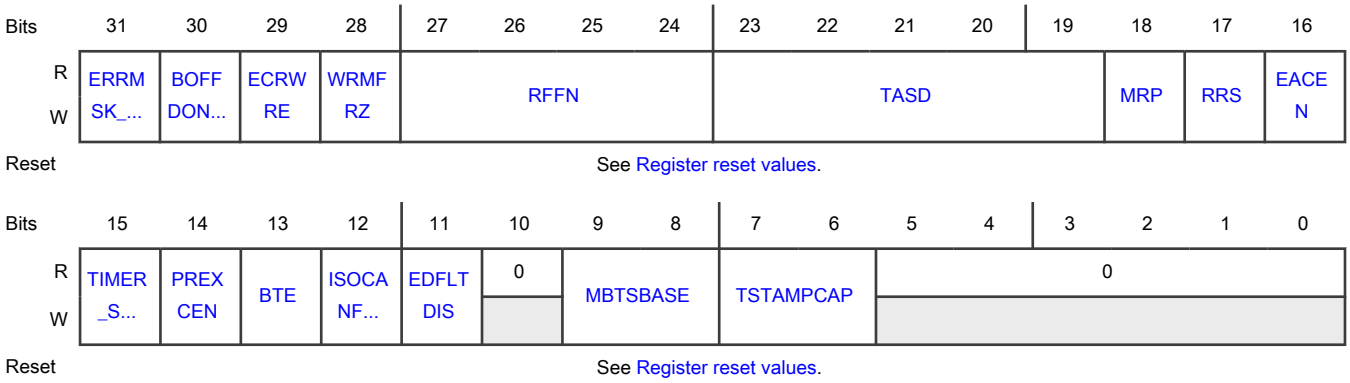
RFFN[3:0]	Number of Legacy Rx FIFO filter elements	Message buffers occupied by Legacy Rx FIFO and ID filter table	Remaining available mailboxes	Legacy Rx FIFO ID filter table elements affected by Rx individual masks	Legacy Rx FIFO ID filter table elements affected by Legacy Rx FIFO global mask
0h	8	MB 0-7	MB 8-95	Elements 0-7	none
1h	16	MB 0-9	MB 10-95	Elements 0-9	Elements 10-15
2h	24	MB 0-11	MB 12-95	Elements 0-11	Elements 12-23
3h	32	MB 0-13	MB 14-95	Elements 0-13	Elements 14-31
4h	40	MB 0-15	MB 16-95	Elements 0-15	Elements 16-39
5h	48	MB 0-17	MB 18-95	Elements 0-17	Elements 18-47
6h	56	MB 0-19	MB 20-95	Elements 0-19	Elements 20-55
7h	64	MB 0-21	MB 22-95	Elements 0-21	Elements 22-63
8h	72	MB 0-23	MB 24-95	Elements 0-23	Elements 24-71
9h	80	MB 0-25	MB 26-95	Elements 0-25	Elements 26-79
Ah	88	MB 0-27	MB 28-95	Elements 0-27	Elements 28-87
Bh	96	MB 0-29	MB 30-95	Elements 0-29	Elements 30-95
Ch	104	MB 0-31	MB 32-95	Elements 0-31	Elements 32-103

Table continues on the next page...

Table 439. Legacy Rx FIFO filter: possible structures (continued)

RFFN[3:0]	Number of Legacy Rx FIFO filter elements	Message buffers occupied by Legacy Rx FIFO and ID filter table	Remaining available mailboxes	Legacy Rx FIFO ID filter table elements affected by Rx individual masks	Legacy Rx FIFO ID filter table elements affected by Legacy Rx FIFO global mask
Dh	112	MB 0-33	MB 34-95	Elements 0-31	Elements 32-111
Eh	120	MB 0-35	MB 36-95	Elements 0-31	Elements 32-119
Fh	128	MB 0-37	MB 38-95	Elements 0-31	Elements 32-127

Diagram



Register reset values

Register	Reset value
CTRL2	CAN_0: 0060_0000h CAN_1,CAN_2: 0080_0000h CAN_3–CAN_5: 00A0_0000h

Fields

Field	Function
31 ERRMSK_FAST	Error Interrupt Mask for errors detected in the data phase of fast CAN FD frames This bit provides a mask for the ERRINT_FAST interrupt in ESR1 register. 0b - ERRINT_FAST error interrupt disabled. 1b - ERRINT_FAST error interrupt enabled.
30 BOFFDONEMSK	Bus Off Done Interrupt Mask This bit provides a mask for the bus off done interrupt in ESR1 register. 0b - Bus off done interrupt disabled. 1b - Bus off done interrupt enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
29 ECRWRE	Error-correction Configuration Register Write Enable This bit enables the MECR register to be updated. This bit is automatically set to 0 if the protocol described in Detection and correction of memory errors is not followed. 0b - Disable update. 1b - Enable update.
28 WRMFRZ	Write-Access To Memory In Freeze Mode Enable unrestricted write access to FlexCAN memory in Freeze mode. This bit can only be written in Freeze mode and has no effect out of Freeze mode. MCR[RFEN] must not be set during FlexCAN memory initialization. 0b - Maintain the write access restrictions. 1b - Enable unrestricted write access to FlexCAN memory.
27-24 RFFN	Number Of Legacy Rx FIFO Filters This 4-bit field defines the number of Rx Legacy FIFO filters, as shown in Table 439. The maximum selectable number of filters is determined by the chip. This field can only be written in Freeze mode as it is blocked by hardware in other modes. This field must not be programmed with values that cause the number of message buffers occupied by Legacy Rx FIFO and Legacy Rx FIFO ID Filter to exceed the number of mailboxes present, as defined by MCR[MAXMB]. **NOTE** Each group of eight filters occupies a memory space equivalent to two message buffers, which means that the more filters are implemented the fewer mailboxes will be available. Considering that the Legacy Rx FIFO occupies the memory space originally reserved for MB0-5, RFFN should be programmed with a value corresponding to a number of filters not greater than the number of available memory words, which can be calculated as follows: (SETUP_MB - 6) × 4 where SETUP_MB is the smaller of the parameter NUMBER_OF_MB and MCR[MAXMB]. The number of remaining mailboxes available will be: (SETUP_MB - 8) - (RFFN × 2) If the number of Legacy Rx FIFO filters programmed through RFFN exceeds the SETUP_MB value (memory space available), then the exceeding ones will not be functional. **NOTE** • The number of the last remaining available mailboxes is defined by the smaller of NUMBER_OF_MB minus 1 and MCR[MAXMB]. • If Rx Individual Mask registers are not enabled then all Legacy Rx FIFO filters are affected by the Legacy Rx FIFO Global Mask.
23-19	Tx Arbitration Start Delay

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>TASD</td><td>This 5-bit field indicates how many CAN bits the Tx arbitration process start point can be delayed from the first bit of CRC field on CAN bus. See Tx arbitration start delay for more details. This field can be written only in Freeze mode because it is blocked by hardware in other modes.</td></tr>
<tr><td>18
MRP</td><td>Mailboxes Reception Priority
If this bit is set the matching process starts from the mailboxes and if no match occurs the matching continues on the Legacy Rx FIFO or Enhanced RX FIFO . This bit can be written only in Freeze mode because it is blocked by hardware in other modes.
NOTE
The descriptions of the field settings vary by module instance.
<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>CAN_0</td><td>0b - Matching starts from Legacy Rx FIFO or Enhanced Rx FIFO and continues on mailboxes.
1b - Matching starts from mailboxes and continues on Legacy Rx FIFO or Enhanced Rx FIFO.</td></tr><tr><td>CAN_1</td><td>0b - Matching starts from Rx FIFO and continues on mailboxes.
1b - Matching starts from mailboxes and continues on Rx FIFO.</td></tr><tr><td>CAN_2</td><td>0b - Matching starts from Rx FIFO and continues on mailboxes.
1b - Matching starts from mailboxes and continues on Rx FIFO.</td></tr><tr><td>CAN_3</td><td>0b - Matching starts from Rx FIFO and continues on mailboxes.
1b - Matching starts from mailboxes and continues on Rx FIFO.</td></tr><tr><td>CAN_4</td><td>0b - Matching starts from Rx FIFO and continues on mailboxes.
1b - Matching starts from mailboxes and continues on Rx FIFO.</td></tr><tr><td>CAN_5</td><td>0b - Matching starts from Rx FIFO and continues on mailboxes.
1b - Matching starts from mailboxes and continues on Rx FIFO.</td></tr></table></td></tr>
<tr><td>17
RRS</td><td>Remote Request Storing
If this bit is asserted a remote request frame is submitted to a matching process and stored in the corresponding message buffer in the same fashion as a data frame. No automatic remote response frame will be generated.
If this bit is negated the remote request frame is submitted to a matching process and an automatic remote response frame is generated if a message buffer with CODE = 1010b is found with the same ID.
This bit can be written only in Freeze mode because it is blocked by hardware in other modes.
0b - Remote response frame is generated.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Remote request frame is stored.
16 EACEN	Entire Frame Arbitration Field Comparison Enable For Rx Mailboxes This bit controls the comparison of IDE and RTR bits within Rx mailbox filters with their corresponding bits in the incoming frame by the matching process. This bit does not affect matching for Legacy Rx FIFO or Enhanced Rx FIFO. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0b - Rx mailbox filter's IDE bit is always compared and RTR is never compared despite mask bits. 1b - Enables the comparison of both Rx mailbox filter's IDE and RTR bit with their corresponding bits within the incoming frame. Mask bits do apply.
15 TIMER_SRC	Timer Source Selects the time tick source used for incrementing the free running timer counter. This bit can be written in Freeze mode only. 0b - The free running timer is clocked by the CAN bit clock, which defines the baud rate on the CAN bus. 1b - The free running timer is clocked by an external time tick. The period can be either adjusted to be equal to the baud rate on the CAN bus, or a different value as required. See the device-specific section for details about the external time tick.
14 PREXCEN	Protocol Exception Enable This bit enables the protocol exception feature. This field is writable only in Freeze mode. **NOTE** See Protocol exception event in the CAN Protocol standard (ISO 11898-1) for details. 0b - Protocol exception is disabled. 1b - Protocol exception is enabled.
13 BTE	Bit Timing Expansion enable This field enables the use of the EPRS, EDCBT and ENCBT registers to configure the CAN bit timing segments, instead of using the bit timing fields of CBT, FDCBT, and CTRL1 registers. If CTRL2[BTE] is set: • PRESDIV, PROPSEG, PSEG1, PSEG2, and RJW fields in CTRL1 register are read as zero, and a write operation in them has no effect. • EPRESDIV, ERJW, EPROPSEG, EPSEG1, and EPSEG2 fields in the CBT register, and the FDCBT register, are read as zero and a write operation to them has no effect. • The ETDCOFF, ETDCEN, ETDCFAIL, and ETDCVAL fields of the ETDC register are used by FlexCAN instead of TDCOFF, TDCEN, TDCFAIL, and TDCVAL fields in FDCTRL register, which are read as zero, and a write operation in them has no effect. • ETDC[TDMDIS] can be used for disabling transceiver delay measurement.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - CAN Bit timing expansion is disabled. 1b - CAN bit timing expansion is enabled.
12 ISOCANFDEN	ISO CAN FD Enable This field enables the CAN FD protocol according to ISO specification (ISO 11898-1) (see CAN FD ISO compliance). This field is writable only in Freeze mode. **NOTE** FlexCAN is able to transmit FD frame format according to CAN Protocol standard (ISO 11898-1). 0b - FlexCAN operates using the non-ISO CAN FD protocol. 1b - FlexCAN operates using the ISO CAN FD protocol (ISO 11898-1).
11 EDFLTDIS	Edge Filter Disable This bit disables the edge filter used during the bus integration state. When the Edge Filter is enabled, two consecutive nominal time quanta with dominant bus state are required to detect an edge that causes synchronization. When synchronization occurs, the counting of the sequence of eleven consecutive recessive bits is restarted. The edge filter prevents the dominant pulses that are shorter than a nominal bit time (present during the data phase of an FD frame) from being mistaken for an idle condition. This field is writable only in Freeze mode. **NOTE** See Bus Integration state in the CAN Protocol standard (ISO 11898-1) for details. 0b - Edge filter is enabled 1b - Edge filter is disabled
10 —	Reserved
9-8 MBTSBASE	Message Buffer Time Stamp Base This field selects which time base is used for capturing the 16-bit TIME_STAMP field of the message buffer register. There are three time base options: 1. TIMER 2. Lower 16 bits of high resolution time base 3. Upper 16 bits of high resolution time base This field can be written in Freeze Mode only.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Message buffer time stamp base is TIMER 01b - Message buffer time stamp base is lower 16 bits of high resolution timer 10b - Message buffer time stamp base is upper 16 bits of high resolution timer 11b - Reserved
7-6 TSTAMPCAP	Time Stamp Capture Point This field configures the point in time when a 32-bit time base is captured during a CAN frame and stored in the high resolution time stamp register (HR_TIME_STAMP). For classical CAN frames, capture points can be the start of frame bit or the point in time a CAN frame is considered valid, which is the 7th bit of end of frame for transmission and the 6th bit of the end of frame for reception. For CAN FD frames, the high resolution time stamp can be captured at the start of frame, the point in time a CAN FD frame is considered valid, or the res bit. This field can be written in Freeze mode only. 00b - The high resolution time stamp capture is disabled 01b - The high resolution time stamp is captured in the end of the CAN frame 10b - The high resolution time stamp is captured in the start of the CAN frame 11b - The high resolution time stamp is captured in the start of frame for classical CAN frames and in res bit for CAN FD frames
5-0 —	Reserved

70.6.2.15 Error and Status 2 Register (ESR2)

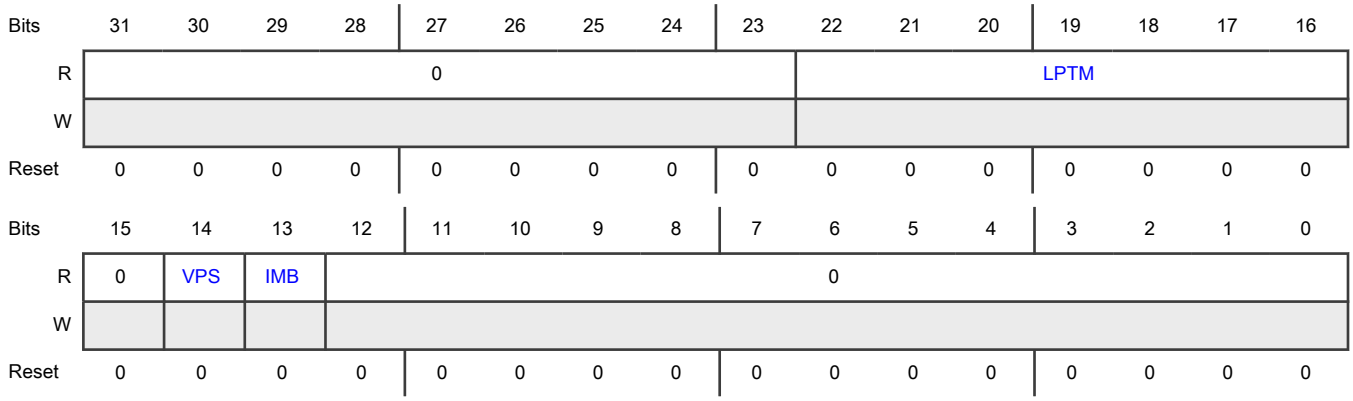
Offset

Register	Offset
ESR2	38h

Function

This register reports some general status information.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 LPTM	Lowest Priority Tx Mailbox If ESR2[VPS] is asserted, this field indicates the lowest number inactive mailbox (see the ESR2[IMB] bit description). If there is no inactive mailbox then the mailbox indicated depends on the value of CTRL1[LBUF]. If CTRL1[LBUF] is negated, then the mailbox indicated is the one that has the greatest arbitration value (see Highest-priority mailbox first). If CTRL1[LBUF] is asserted then the mailbox indicated is the highest number active Tx mailbox. If a Tx mailbox is being transmitted it is not considered in LPTM calculation. If ESR2[IMB] is not asserted and a frame is transmitted successfully, LPTM is updated with its mailbox number.
15 —	Reserved
14 VPS	Valid Priority Status This bit indicates whether ESR2[IMB] and ESR2[LPTM] contents are currently valid or not. It is asserted upon every complete Tx arbitration process unless the CPU writes to Control and Status word of a mailbox that has already been scanned, that is, it is behind Tx Arbitration Pointer, during the Tx arbitration process. If there is no inactive mailbox and only one Tx mailbox that is being transmitted then VPS is not asserted. This bit is negated upon the start of every Tx arbitration process or upon a write to Control and Status word of any mailbox. **NOTE** ESR2[VPS] is not affected by any CPU write into Control Status (C/S) of an MB that is blocked by abort mechanism. When MCR[AEN] is asserted, the abort code write in C/S of an MB that is being transmitted (pending abort), or any write attempt into a Tx MB with IFLAG set is blocked. 0b - Contents of IMB and LPTM are invalid. 1b - Contents of IMB and LPTM are valid.

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 IMB	Inactive Mailbox If ESR2[VPS] is asserted, this bit indicates whether there is any inactive mailbox (CODE field is either 1000b or 0b). This bit is asserted in the following cases: • During arbitration, if an ESR2[LPTM] is found and it is inactive. • If ESR2[IMB] is not asserted and a frame is transmitted successfully. This bit is always cleared in start of arbitration (see Arbitration process). **NOTE** ESR2[LPTM] mechanism has the following behavior: if an MB is successfully transmitted and ESR2[IMB]=0 (no inactive mailbox), then ESR2[VPS] and ESR2[IMB] are asserted and the index related to the MB just transmitted is loaded into ESR2[LPTM]. 0b - If ESR2[VPS] is asserted, the ESR2[LPTM] is not an inactive mailbox. 1b - If ESR2[VPS] is asserted, there is at least one inactive mailbox. LPTM content is the number of the first one.
12-0 —	Reserved

70.6.2.16 CRC Register (CRCR)

Offset

Register	Offset
CRCR	44h

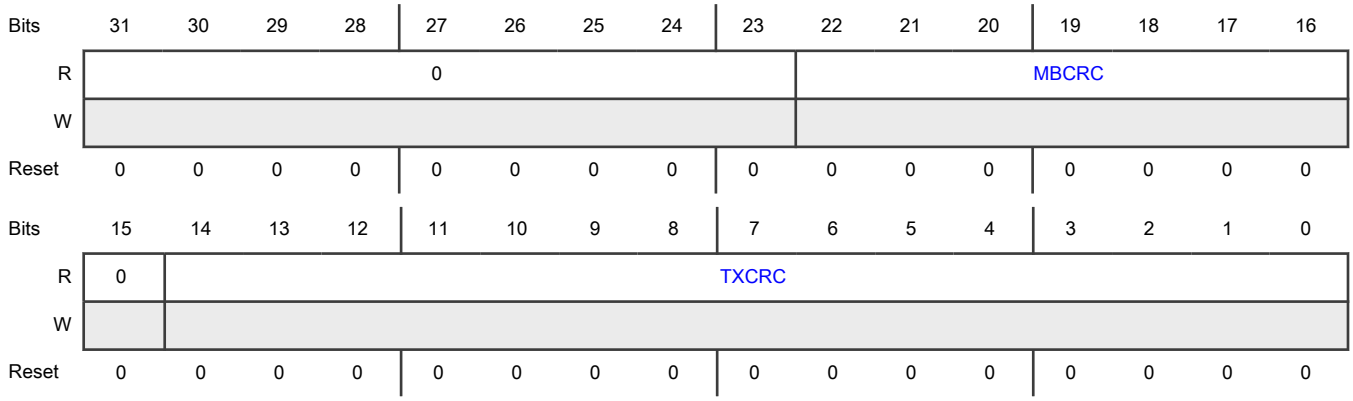
Function

This register provides information about the CRC of transmitted messages for non-FD messages. This register only reports the 15 low order bits of CRC calculations for messages in CAN FD format that require either 17 or 21 bits. For CAN FD format frames, the FDCRC register must be used. This register is updated at the same time the Tx interrupt flag is asserted.

NOTE

See CRC sequence calculation in the CAN Protocol standard (ISO 11898-1) for details.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 MBCRC	CRC Mailbox This field indicates the number of the mailbox corresponding to the value in CRCR[TXCRC] field.
15 —	Reserved
14-0 TXCRC	Transmitted CRC value This field indicates the CRC value of the last transmitted message for non-FD frames. For FD frames, CRC value is reported in FDCRC register.

70.6.2.17 Legacy Rx FIFO Global Mask Register (RXFGMASK)

Offset

Register	Offset
RXFGMASK	48h

Function

This register is located in RAM.

If Legacy Rx FIFO is enabled, RXFGMASK is used to mask the Legacy Rx FIFO ID filter table elements that do not have a corresponding RXIMR according to CTRL2[RFFN] field setting.

This register can only be written in Freeze mode as it is blocked by hardware in other modes.

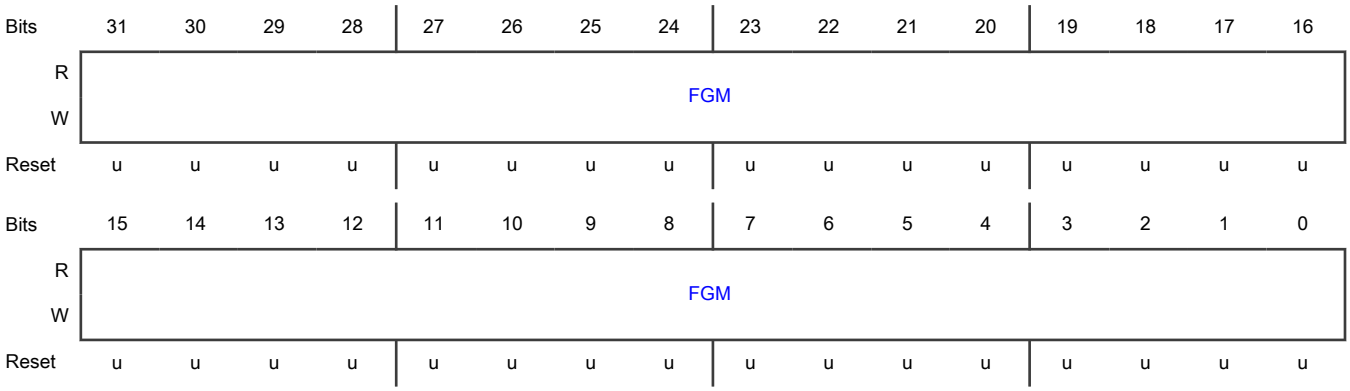
The following table shows how the FGM bits correspond to each IDAF field.

Table 440. Correspondence of Legacy Rx FIFO global mask bits to IDF fields

Legacy Rx FIFO ID filter table elements format (MCR[IDAM])	Identifier acceptance filter fields					
	RTR	IDE	RXIDA	RXIDB [1]	RXIDC [2]	Reserved
A	FGM[31]	FGM[30]	FGM[29:1]	—	—	FGM[0]
B	FGM[31], FGM[15]	FGM[30], FGM[14]	—	FGM[29:16], FGM[13:0]		—
C	—	—		—	FGM[31:24], FGM[23:16], FGM[15:8], FGM[7:0]	

1. If MCR[IDAM] is equivalent to format B, only the fourteen most significant bits of the identifier of the incoming frame are compared with the Legacy Rx FIFO filter.
2. If MCR[IDAM] is equivalent to format C, only the eight most significant bits of the identifier of the incoming frame are compared with the Legacy Rx FIFO filter.

Diagram



Fields

Field	Function
31-0 FGM	Legacy Rx FIFO Global Mask Bits These bits mask the ID filter table elements bits in a perfect alignment. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

70.6.2.18 Legacy Rx FIFO Information Register (RXFIR)

Offset

Register	Offset
RXFIR	4Ch

Function

RXFIR provides information on Legacy Rx FIFO.

This register is the port through which the CPU accesses the output of the Legacy RXFIR FIFO located in RAM. The Legacy RXFIR FIFO is written by the FlexCAN whenever a new message is moved into the Legacy Rx FIFO. Also, its output is updated whenever the output of the Legacy Rx FIFO is updated with the next message. See Legacy Rx FIFO for instructions on reading this register.

NOTE

RXFIR can be written only during memory initialization, due to the error code correction (ECC) feature. In every other case the register is read-only.

Diagram

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

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

Fields

Field	Function
31-9 —	Reserved
8-0 IDHIT	Identifier Acceptance Filter Hit Indicator This field indicates which Identifier Acceptance filter was hit by the received message that is in the output of the Legacy Rx FIFO. If multiple filters match the incoming message ID then the first matching IDAF found (lowest number) by the matching process is indicated. This field is valid only when IFLAG1[BUF5I] is asserted.

70.6.2.19 CAN Bit Timing Register (CBT)

Offset

Register	Offset
CBT	50h

Function

This register is an alternative way to store the CAN bit timing variables described in CTRL1 register. EPRESDIV, EPROPSEG, EPSEG1, EPSEG2, and ERJW are extended versions of PRESDIV, PROPSEG, PSEG1, PSEG2, and RJW fields respectively.

The BTF bit selects the use of the timing variables defined in this register.

The contents of this register are not affected by soft reset.

NOTE

The CAN bit variables in CTRL1 and in CBT are stored in the same register.

NOTE

When the CAN FD feature is enabled (MCR[FDEN] is set), always set CBT[BTF].

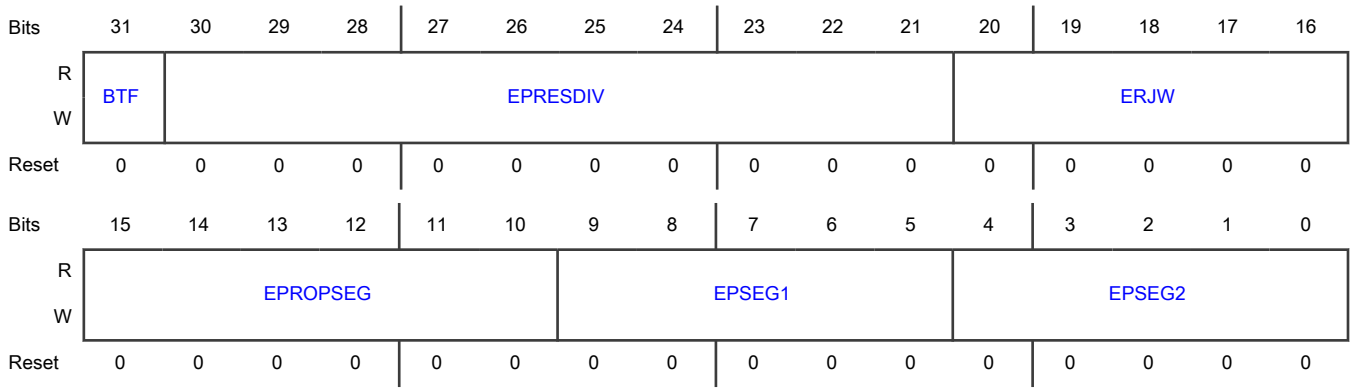
NOTE

The user must ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1).

NOTE

If CTRL2[BTE] is set, EPRESDIV, ERJW, EPROPSEG, EPSEG1, and EPSEG2 fields are read as zero, and a write operation into them has no effect.

Diagram



Fields

Field	Function
31 BTF	Bit Timing Format Enable Enables the use of extended CAN bit timing fields EPRESDIV, EPROPSEG, EPSEG1, EPSEG2. and ERJW replacing the CAN bit timing variables defined in CTRL1 register. This field can be written in Freeze mode only. 0b - Extended bit time definitions disabled. 1b - Extended bit time definitions enabled.
30-21 EPRESDIV	Extended Prescaler Division Factor This 10-bit field defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency when CBT[BTF] is asserted, otherwise it has no effect. It extends the CTRL1[PRESDIV] value range. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency (see Protocol timing). This field can be written only in Freeze mode because it is blocked by hardware in other modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Sclock frequency = PE clock frequency / (EPRESDIV + 1)
20-16 ERJW	Extended Resync Jump Width This 5-bit field defines the maximum number of time quanta that a bit time can be changed by one re-synchronization when CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CTRL1[RJW] value range. One time quantum is equal to the Sclock period. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Resync Jump Width = ERJW + 1.
15-10 EPROPSEG	Extended Propagation Segment This 6-bit field defines the length of the propagation segment in the bit time when CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CTRL1[PROPSEG] value range. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Propagation Segment Time = (EPROPSEG + 1) × Time-Quanta. Time-Quantum = one Sclock period.
9-5 EPSEG1	Extended Phase Segment 1 This 5-bit field defines the length of phase segment 1 in the bit time when CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CTRL1[PSEG1] value range. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 1 = (EPSEG1 + 1) × Time-Quanta. Time-Quantum = one Sclock period.
4-0 EPSEG2	Extended Phase Segment 2 This 5-bit field defines the length of phase segment 2 in the bit time when CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CTRL1[PSEG2] value range. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 1 = (EPSEG2 + 1) × Time-Quanta. Time-Quantum = one Sclock period.

70.6.2.20 Interrupt Masks 3 Register (IMASK3)

Offset

Register	Offset
IMASK3	6Ch

Function

This register allows any number of a range of the 32 message buffer interrupts to be enabled or disabled for MB95 to MB64. It contains one interrupt mask bit per buffer, enabling the CPU to determine which buffer generates an interrupt after a successful transmission or reception, that is, when the corresponding IFLAG3 bit is set.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
CAN_0	IMASK3	—
CAN_1	—	IMASK3
CAN_2	—	IMASK3
CAN_3	—	IMASK3
CAN_4	—	IMASK3
CAN_5	—	IMASK3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BUF95TO64M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BUF95TO64M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BUF95TO64M	Buffer MBi Mask Each bit enables or disables the corresponding FlexCAN message buffer interrupt for MB95 to MB64. When CAN FD is enabled, the MB range is defined in accordance with the MBDSRs bit fields of FDCTRL register. **NOTE** Setting or clearing a bit in the IMASK3 Register can assert or negate an interrupt request, if the corresponding IFLAG3 bit is set. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

70.6.2.21 Interrupt Flags 3 Register (IFLAG3)

Offset

Register	Offset
IFLAG3	74h

Function

This register defines the flags for the 32 message buffer interrupts for MB95 to MB64. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding IFLAG3 bit. If the corresponding IMASK3 bit is set, an interrupt will be generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect.

Before updating MCR[MAXMB] field, CPU must service the IFLAG3 bits whose MB value is greater than the MAXMB to be updated; otherwise, they will remain set and be inconsistent with the number of MBs available.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
CAN_0	IFLAG3	—
CAN_1	—	IFLAG3
CAN_2	—	IFLAG3
CAN_3	—	IFLAG3
CAN_4	—	IFLAG3
CAN_5	—	IFLAG3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BUF95TO64															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BUF95TO64															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BUF95TO64	Buffer MBi Interrupt Each bit flags the corresponding FlexCAN message buffer interrupt for MB95 to MB64.When CAN FD is enabled, the MB range is defined in accordance to the MBDSRs bit fields of FDCTRL register. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.

70.6.2.22 Rx Individual Mask Registers (RXIMR0 - RXIMR95)

Offset

For n = 0 to 95:

Register	Offset
RXIMRn	880h + (n × 4h)

Function

The RX Individual Mask registers are used to store the acceptance masks for ID filtering in Rx MBs and the Legacy Rx FIFO.

When the Legacy Rx FIFO is disabled (MCR[RFEN] bit is negated), an individual mask is provided for each available Rx mailbox on a one-to-one correspondence. When the Legacy Rx FIFO is enabled (MCR[RFEN] bit is asserted), an individual mask is provided for each Legacy Rx FIFO ID filter table element on a one-to-one correspondence depending on the setting of CTRL2[RFFN] (see Legacy Rx FIFO).

RXIMR0 stores the individual mask associated with either MB0 or ID filter table element 0, RXIMR1 stores the individual mask associated with either MB1 or ID filter table element 1, and so on.

RXIMR registers can only be accessed by the CPU when the module is in Freeze mode; otherwise, they are blocked by hardware. These registers are not affected by reset. They are located in RAM and must be explicitly initialized prior to any reception.

It is possible for the RXIMR memory region to be accessed as general purpose memory. See Bus interface for more information.

NOTE

Each module instance supports a different number of registers.

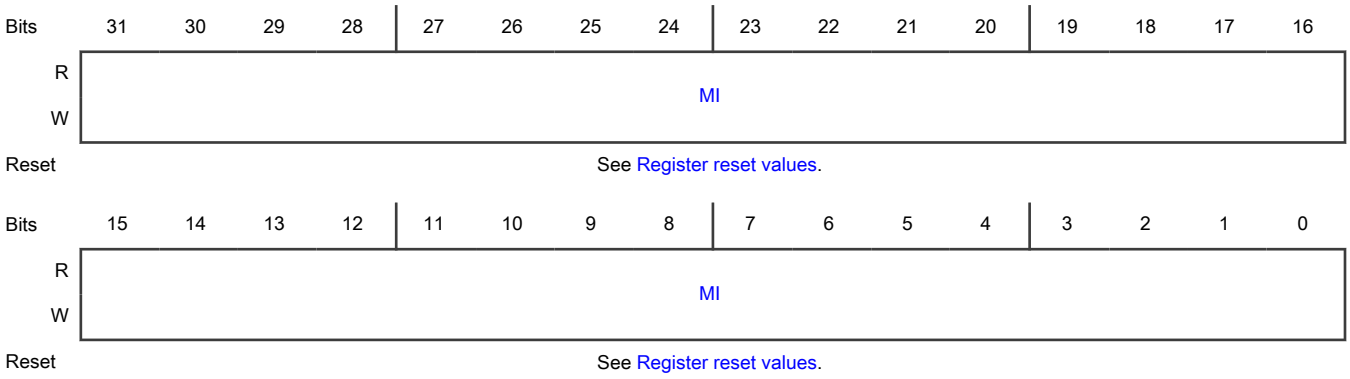
Instance	Register supported	Register not supported
CAN_0	RXIMR0–RXIMR95	—
CAN_1	RXIMR0–RXIMR63	RXIMR64–RXIMR95
CAN_2	RXIMR0–RXIMR63	RXIMR64–RXIMR95
CAN_3	RXIMR0–RXIMR31	RXIMR32–RXIMR95
CAN_4	RXIMR0–RXIMR31	RXIMR32–RXIMR95

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
CAN_5	RXIMR0–RXIMR31	RXIMR32–RXIMR95

Diagram



Register reset values

Register	Reset value
RXIMR0–RXIMR31	CAN_0–CAN_5: undefined
RXIMR32–RXIMR63	CAN_0–CAN_2: undefined CAN_3–CAN_5: Register not supported
RXIMR64–RXIMR95	CAN_0: undefined CAN_1–CAN_5: Register not supported

Fields

Field	Function
31-0 MI	Individual Mask Bits Each individual mask bit masks the corresponding bit in both the mailbox filter and Legacy Rx FIFO ID filter table element in distinct ways. For mailbox filters, see the RXMGMASK register description. For Legacy Rx FIFO ID filter table elements, see the RXFGMASK register description. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

70.6.2.23 Memory Error Control Register (MECR)

Offset

Register	Offset
MECR	AE0h

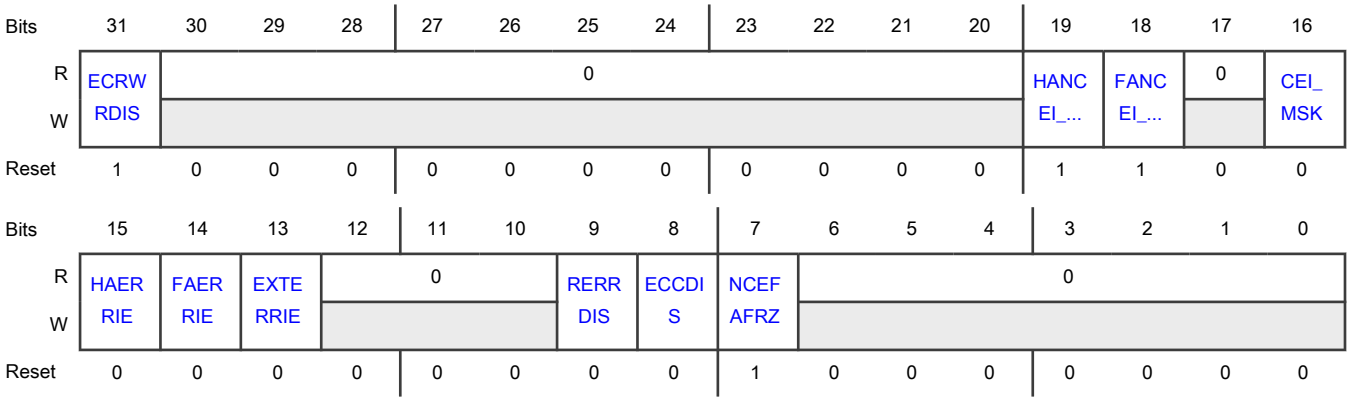
Function

This register contains control bits for memory error detection and correction (ECC).

NOTE

When bit CTRL2[ECRWRE] is 0, writes in this register are blocked, except in the ECRWRDIS bit.

Diagram



Fields

Field	Function
31 ECRWRDIS	Error Configuration Register Write Disable Disables writes on this register. This bit is automatically set to 1 (disabled) when CTRL2[ECRWRE] is enabled. The protocol described in section Detection and correction of memory errors must be followed. 0b - Write is enabled. 1b - Write is disabled.
30-20 —	Reserved
19 HANCEI_MSK	Host Access With Non-Correctable Errors Interrupt Mask Enables the interrupt in case of non-correctable errors detected in memory reads issued by the host (CPU). 0b - Interrupt is disabled. 1b - Interrupt is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 FANCEI_MSK	FlexCAN Access With Non-Correctable Errors Interrupt Mask Enables the interrupt in case of non-correctable errors detected in memory reads issued by the FlexCAN internal processes. 0b - Interrupt is disabled. 1b - Interrupt is enabled.
17 —	Reserved
16 CEI_MSK	Correctable Errors Interrupt Mask Enables the interrupt in case of correctable errors detected in memory reads issued by the host or FlexCAN internal processes. 0b - Interrupt is disabled. 1b - Interrupt is enabled.
15 HAERRIE	Host Access Error Injection Enable Enables the injection of errors only in memory reads issued by the host (CPU). 0b - Injection is disabled. 1b - Injection is enabled.
14 FAERRIE	FlexCAN Access Error Injection Enable Enables the injection of errors only in memory reads issued by the FlexCAN internal processes. 0b - Injection is disabled. 1b - Injection is enabled.
13 EXTERRIE	Extended Error Injection Enable Memory accesses performed by internal FlexCAN processes are 64-bit. This bit extends the error injection on 32-bit memory accesses to the complementary 32-bit word using the same 32-bit error injection data and parity words. See Error Injection Data Pattern Register (ERRIDPR) and Error Injection Parity Pattern Register (ERRIPPR). 0b - Error injection is applied only to the 32-bit word. 1b - Error injection is applied to the 64-bit word.
12-10 —	Reserved
9 RERRDIS	Error Report Disable Disables the update of the error report registers. The update of error-related flags and the generation of bus transfer errors are still active.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE When reading the report registers, this bit must be set to assure coherence on the consecutive register reads. 0b - Enable updates of the error report registers. 1b - Disable updates of the error report registers.
8 ECCDIS	Error Correction Disable Disables completely the memory detection and correction mechanism. Besides disabling the error report mechanism, it also stops the update of the error-related flags and generation of bus transfer errors. The parity bits continue being calculated and written into memory on write transactions. 0b - Enable memory error correction. 1b - Disable memory error correction.
7 NCEFAFRZ	Non-Correctable Errors In FlexCAN Access Put Device In Freeze Mode Determines the response when a non-correctable error is detected in a memory read performed by FlexCAN internal processes. In this case, entering Freeze mode prevents corrupted data from being treated as valid by FlexCAN internal processes. 0b - Keep normal operation. 1b - Put FlexCAN in Freeze mode (see section "Freeze mode").
6-0 —	Reserved

70.6.2.24 Error Injection Address Register (ERRIAR)

Offset

Register	Offset
ERRIAR	AE4h

Function

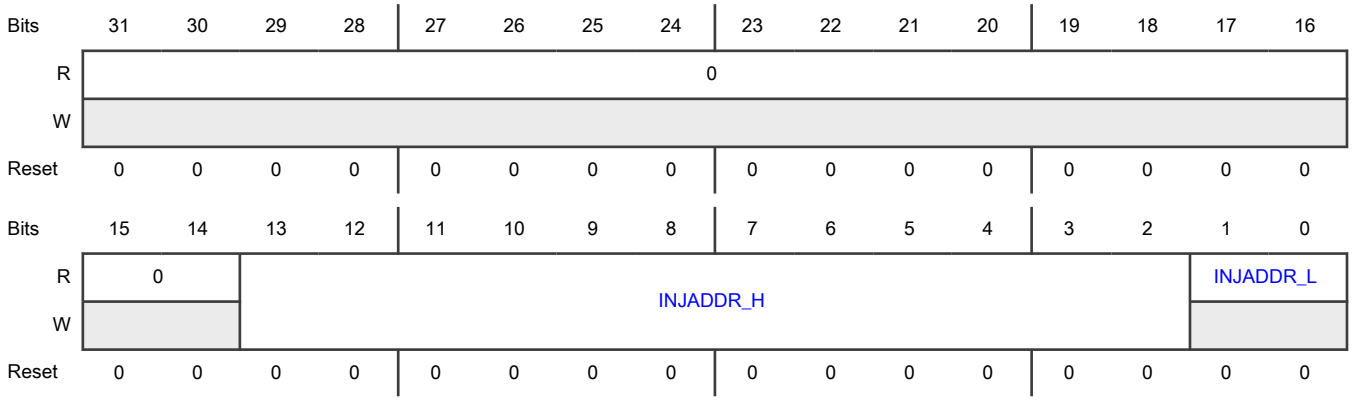
This register holds the address where error is to be injected.

See the chip-specific FlexCAN information for the FlexCAN RAM to memory map address mapping table.

NOTE

For RXFIR, Enhanced RX FIFO, and HR_TIME_STAMP addresses, you must inject ECC errors in host access only.

Diagram



Fields

Field	Function
31-14 —	Reserved
13-2 INJADDR_H	Error Injection Address High This read-write field defines the twelve most significant bits of the physical RAM address where error is to be injected (see table above).
1-0 INJADDR_L	Error Injection Address Low This read-only field defines the two least significant bits of the physical RAM address where error is to be injected. They ensure that the address is on a thirty-two-bit boundary.

70.6.2.25 Error Injection Data Pattern Register (ERRIDPR)

Offset

Register	Offset
ERRIDPR	AE8h

Function

Holds the error pattern to be injected in the data word read from memory.

Diagram

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

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

Fields

Field	Function
31-0 DFLIP	Data flip pattern Bits set to 1 in the flip pattern cause the corresponding data bit in the word read from memory to invert.

70.6.2.26 Error Injection Parity Pattern Register (ERRIPPR)

Offset

Register	Offset
ERRIPPR	AECh

Function

Holds the error pattern to be injected in parity bits read from memory along with data word.

Bits set to 1 in the flip pattern cause the corresponding parity bit in the word read from memory to invert.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			PFLIP3					0			PFLIP2				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			PFLIP1					0			PFLIP0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-24 PFLIP3	Parity Flip Pattern For Byte 3 (most significant)
23-21 —	Reserved
20-16 PFLIP2	Parity Flip Pattern For Byte 2
15-13 —	Reserved
12-8 PFLIP1	Parity Flip Pattern For Byte 1
7-5 —	Reserved
4-0 PFLIP0	Parity Flip Pattern For Byte 0 (Least Significant)

70.6.2.27 Error Report Address Register (RERRAR)

Offset

Register	Offset
RERRAR	AF0h

Function

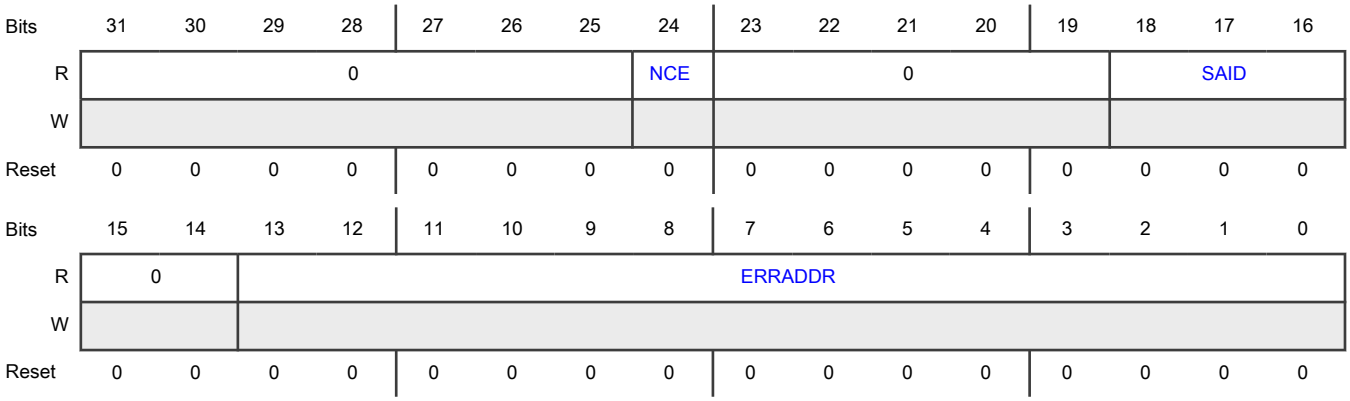
Reports the address used for an access in which an error (correctable or non-correctable) was detected, and also reports the identification of the source of that access.

This address is always reported using a 32-bit alignment. Non-aligned accesses (ERRADDR[1:0] non-0) are reported with the address aligned and data is reported in RERRDR accordingly shifted. In case of errors detected in accesses larger than 32-bit (as performed by FlexCAN internal processes), the address of the 32-bit word which the error was detected is reported. In case of errors detected in more than one 32-bit word, only the least significant address is reported.

Table 441. Source of memory access

SAID[2:0]	Error during...
0	Move-out FlexCAN access
1	Move-in
2	Tx Arbitration
3	Rx Matching
4	Move-out Host access
5-7	Reserved

Diagram



Fields

Field	Function
31-25 —	Reserved
24 NCE	Non-Correctable Error Indicates that the report is due to a non-correctable error. 0b - Reporting a correctable error 1b - Reporting a non-correctable error
23-19 —	Reserved
18-16 SAID	SAID SAID[2] — Identification of the requester of the memory read request: • 0 = Requested by FlexCAN internal processes • 1 = Requested by host (CPU) SAID[1] — Details of FlexCAN operation:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0 = Move • 1 = Scanning SAID[0] — Operation that requested the memory read: • 0 = Transmission • 1 = Reception For more information see Table 441.
15-14 —	Reserved
13-0 ERRADDR	Address Where Error Detected See the description of the Error Injection Address Register (ERRIAR).

70.6.2.28 Error Report Data Register (RERRDR)

Offset

Register	Offset
RERRDR	AF4h

Function

Reports the raw data (unmodified by the correction performed by ECC logic) read from memory with error. The value reported does not represent the transient values of the BUSY bit (see Table 446) when reading a message buffer.

Diagram

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

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

Fields

Field	Function
31-0 RDATA	Raw data word read from memory with error

70.6.2.29 Error Report Syndrome Register (RERRSYNR)

Offset

Register	Offset
RERRSYNR	AF8h

Function

Holds the syndrome detected in a memory read with error. It also reports the bytes which were read in this 32-bit read transaction.

Each SYNDn field indicates the type of error and which bit in byte (n) is affected by the error. (SYND3 corresponds to the most significant byte in the data word read from memory; SYND0 corresponds to the least significant.)

Table 442. Syndrome definition

SYNDn (hex)	Type	Bit affected
00	—	none (no error)
01	Code	0
02	Code	1
04	Code	2
07	Data	5
08	Code	3
0E	Data	7
10	Code	4
13	Data	2
15	Data	6
16	Data	1
19	Data	3
1A	Data	4
1C	Data	0
06	—	All-zeros non-correctable error
1F	—	All-ones non-correctable error
All others	—	Non-correctable error

Each BEn field indicates which byte in the 32-bit word reported was effectively read. The syndrome bits are calculated for all bytes, even for the non-read ones. Errors detected in non-read bytes are indicated (see Error indication) and reported (see Error reporting).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BE3	0			SYND3				BE2	0			SYND2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BE1	0			SYND1				BE0	0			SYND0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BE3	Byte Enabled For Byte 3 (most significant) 0b - The byte was not read. 1b - The byte was read.
30-29 —	Reserved
28-24 SYND3	Error Syndrome For Byte 3 (most significant) See Table 442.
23 BE2	Byte Enabled For Byte 2 0b - The byte was not read. 1b - The byte was read.
22-21 —	Reserved
20-16 SYND2	Error Syndrome For Byte 2 See Table 442.
15 BE1	Byte Enabled For Byte 1 0b - The byte was not read. 1b - The byte was read.
14-13	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
12-8 SYND1	Error Syndrome for Byte 1 See Table 442.
7 BE0	Byte Enabled For Byte 0 (least significant) 0b - The byte was not read. 1b - The byte was read.
6-5 —	Reserved
4-0 SYND0	Error Syndrome For Byte 0 (least significant) See Table 442.

70.6.2.30 Error Status Register (ERRSR)

Offset

Register	Offset
ERRSR	AFCh

Function

Holds the status bits of the error correction and detection operations. These flags can be cleared by writing 1 to them. Writing 0 has no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												HANCEIF	FANCEIF	0	CEIF
W													W1C	W1C		W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												HANCEIOF	FANCEIOF	0	CEIOF
W													W1C	W1C		W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 HANCEIF	Host Access With Non-Correctable Error Interrupt Flag Indicates that a non-correctable error was detected in a memory read initiated by host. A bus transfer error is asserted for that access. If MECR[HANCEI_MSK] is set, the interrupt is asserted. 0b - No non-correctable errors were detected in host accesses so far. 1b - A non-correctable error was detected in a host access.
18 FANCEIF	FlexCAN Access With Non-Correctable Error Interrupt Flag Indicates that a non-correctable error was detected in a memory read initiated by FlexCAN internal processes. If MECR[FANCEI_MSK] is set, the interrupt is asserted. 0b - No non-correctable errors were detected in FlexCAN accesses so far. 1b - A non-correctable error was detected in a FlexCAN access.
17 —	Reserved
16 CEIF	Correctable Error Interrupt Flag Indicates that a correctable error was detected in a memory read. If MECR[CEI_MSK] is set, the interrupt is asserted. 0b - No correctable errors were detected so far. 1b - A correctable error was detected.
15-4 —	Reserved
3 HANCEIOF	Host Access With Non-Correctable Error Interrupt Overrun Flag Indicates that a non-correctable error was detected in a memory read initiated by host when HANCEIF was set. No interrupt is associated with this flag. See Error indication. 0b - No overrun on non-correctable errors in host access 1b - Overrun on non-correctable errors in host access
2 FANCEIOF	FlexCAN Access With Non-Correctable Error Interrupt Overrun Flag Indicates that a non-correctable error was detected in a memory read initiated by FlexCAN internal processes when FANCEIF was set. No interrupt is associated to this flag. See Error indication. 0b - No overrun on non-correctable errors in FlexCAN access 1b - Overrun on non-correctable errors in FlexCAN access
1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 CEIOF	Correctable Error Interrupt Overrun Flag Indicates that a correctable error was detected in a memory read when CEIF was set. No interrupt is associated to this flag. See Error indication. 0b - No overrun on correctable errors 1b - Overrun on correctable errors

70.6.2.31 Enhanced CAN Bit Timing Prescalers (EPRS)

Offset

Register	Offset
EPRS	BF0h

Function

This register defines the CAN bit timing prescalers for the nominal phase and data phase when CTRL2[BTE] is set.

This register is used by the hardware only if CTRL2[BTE] is set; otherwise a write operation has no effect and all fields are read as zero.

This register can be written only in Freeze mode because it is blocked by hardware in other modes.

The contents of this register are not affected by soft reset.

Diagram

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

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

Fields

Field	Function
31-26	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
25-16 EDPRESDIV	Extended Data Phase Prescaler Division Factor This field defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency in the data phase of a CAN FD message when CTRL2[BTE] is set. The Sclock period defines the time quantum of the CAN FD protocol for the data bit rate. Sclock frequency = PE clock frequency / (EDPRESDIV + 1). **NOTE** To minimize errors when processing FD frames, use the same value for ENPRESDIV and EDPRESDIV. For more details see the first NOTE in section CAN FD frames.
15-10 —	Reserved
9-0 ENPRESDIV	Extended Nominal Prescaler Division Factor This field defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency when CTRL2[BTE] is set; otherwise it reads as zero and a write operation has no effect. The Sclock period defines the time quantum of the CAN protocol in the nominal phase. For the reset value, the Sclock frequency is equal to the PE clock frequency (see Protocol timing). Sclock frequency = PE clock frequency / (ENPRESDIV + 1)

70.6.2.32 Enhanced Nominal CAN Bit Timing (ENCBT)

Offset

Register	Offset
ENCBT	BF4h

Function

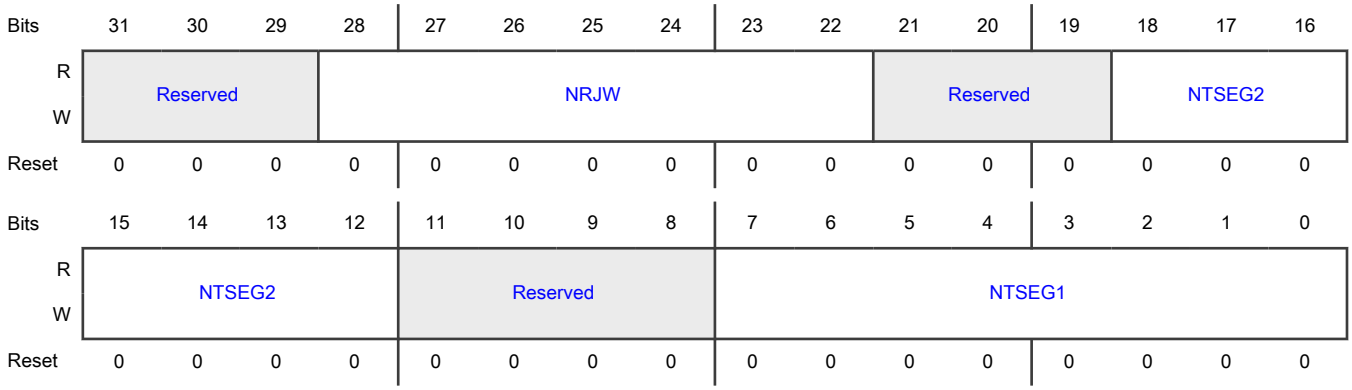
This register provides an alternative way to store the CAN bit timing variables described in the CTRL1 and CBT registers, to get higher CAN bit timing resolution.

This register is used by the hardware only if CTRL2[BTE] is set; otherwise a write operation has no effect and all fields are read as zero.

The contents of this register are not affected by soft reset.

This register can be written only in Freeze mode because it is blocked by hardware in other modes.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-22 NRJW	Nominal Resynchronization Jump Width This field defines the maximum number of time quanta that a nominal bit time can be changed by one resynchronization when CTRL2[BTE] bit is asserted; otherwise it has no effect. One time quantum is equal to the Sclock period. Nominal Resync Jump Width = NRJW + 1.
21-19 —	Reserved
18-12 NTSEG2	Nominal Time Segment 2 This field defines the length of Time Segment 2 in the nominal bit time when CTRL2[BTE] is set; otherwise it has no effect. Nominal Time Segment 2 = (NTSEG2 + 1) × Time-Quanta. Time-Quantum = one Sclock period.
11-8 —	Reserved
7-0 NTSEG1	Nominal Time Segment 1 This field defines the length of Time Segment 1 in the bit time when CTRL2[BTE] is set; otherwise it has no effect. Nominal Time Segment 1 = (NTSEG1 + 1) × Time-Quanta. Time-Quantum = one Sclock period.

70.6.2.33 Enhanced Data Phase CAN bit Timing (EDCBT)

Offset

Register	Offset
EDCBT	BF8h

Function

This register provides an alternative way to store the data phase CAN bit timing variables described in the FDCBT register to achieve higher CAN bit timing resolution.

This register is used by the hardware only if CTRL2[BTE] is set; otherwise a write operation has no effect and all fields are read as zero.

The contents of this register are not affected by soft reset.

This register can be written only in Freeze mode because it is blocked by hardware in other modes.

NOTE

The user must ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1).

NOTE

DTSEG1 must be at least two time quanta.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DTSEG2				Reserved							DTSEG1				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25-22 DRJW	Data Phase Resynchronization Jump Width This field defines the maximum number of time quanta that a data phase bit time can be changed by one resynchronization when CTRL2[BTE] is set; otherwise it has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Data Phase Resync Jump Width = DRJW + 1.
21-16 —	Reserved
15-12 DTSEG2	Data Phase Time Segment 2 This field defines the length of time segment 2 in the data phase bit time when CTRL2[BTE] is set; otherwise it has no effect. Data Phase Time Segment 2 = (DTSEG2 + 1) x Time-Quanta. Time-Quantum = one Sclock period.
11-5 —	Reserved
4-0 DTSEG1	Data Phase Segment 1 This field defines the length of time segment 1 in the data phase bit time when CTRL2[BTE] is set; otherwise it has no effect. Data Phase Time Segment 1 = (NTSEG1 + 1) x Time-Quanta. Time-Quantum = one Sclock period.

70.6.2.34 Enhanced Transceiver Delay Compensation (ETDC)

Offset

Register	Offset
ETDC	BFCh

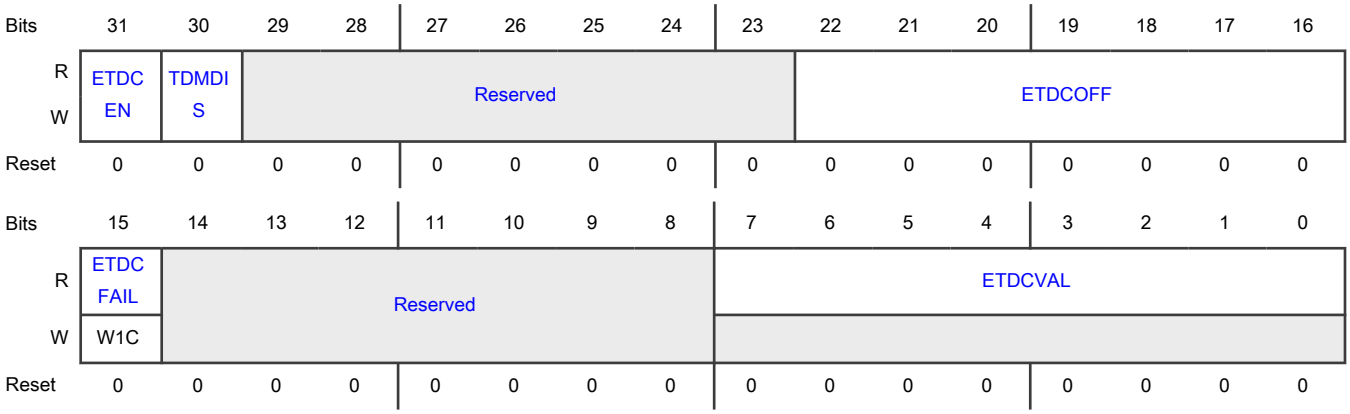
Function

This register contains extended versions of the fields FDCTRL[TDCOFF] and FDCTRL[TDCVAL]. This register is used by the hardware only if CTRL2[BTE] is set; otherwise, a write operation has no effect and all fields are read as zero.

NOTE

See Transmitter delay compensation in the CAN Protocol standard (ISO 11898-1) for details.

Diagram



Fields

Field	Function
31 ETDCEN	Transceiver Delay Compensation Enable This bit can be used for enabling and disabling the TDC feature. It can be written in Freeze mode only. **NOTE** Refer to Transmitter delay compensation in the CAN Protocol standard (ISO 11898-1) for details. **NOTE** TDC must be disabled when the Loop Back Mode is enabled (see CTRL1[LPB] register). 0b - TDC is disabled 1b - TDC is enabled
30 TDMDIS	Transceiver Delay Measurement Disable This bit can be used for disabling the transceiver delay measurement. When the TDC measurement is disabled, the secondary sample point position is determined only by the enhanced TDC offset (ETDCOFF) field. Otherwise, if the TCD measurement is enabled, then the secondary sample point position is determined by the sum of the transceiver delay measurement plus the enhanced TDC offset. This field is not affected by soft reset. **NOTE** This bit can be enabled only if CTRL2[BTE] is set. 0b - TDC measurement is enabled 1b - TDC measurement is disabled
29-23 —	Reserved
22-16 ETDCOFF	Enhanced Transceiver Delay Compensation Offset

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field contains the offset value to be added to the measured transceiver's loop delay in order to define the position of the delayed comparison point when bit rate switching is active. See Transceiver delay compensation for more details on how the loop delay measurement is performed. ETDCOFF can be written in Freeze mode only. Its value can be defined in protocol engine (PE) clock periods (CANCLK, see Protocol timing for more details), and must be selected to be smaller than the CAN bit duration in the data bit rate for proper operation. **NOTE** ETDCOFF must not be configured as zero. **NOTE** If CTRL2[BTE] is set after a chip level hard reset, ETCDOFF is read as 1h.
15 ETDCFAIL	Transceiver Delay Compensation Fail This bit indicates when the Transceiver Delay Compensation (TDC) mechanism is out of range, unable to compensate the transceiver's loop delay and successfully compare the delayed received bits to the transmitted ones (see Transceiver delay compensation). ETDCFAIL sets in the first time FlexCAN detects the out of range condition. The CPU needs to write 1 to clear it. 0b - Measured loop delay is in range. 1b - Measured loop delay is out of range.
14-8 —	Reserved
7-0 ETDCVAL	Enhanced Transceiver Delay Compensation Value This register contains the ETDCOFF field added to the measured value of the transceiver loop delay in the latest transmitted CAN FD frame, with BRS equal to recessive. This field is only updated by the hardware if ETDC[ETDCEN] is set. This field is affected by soft reset. **NOTE** If TDMDIS is set, ETDCVAL stores ETDCOFF only.

70.6.2.35 CAN FD Control Register (FDCTRL)

Offset

Register	Offset
FDCTRL	C00h

Function

This register contains control bits for the CAN FD operation. It also defines the data size of message buffers allocated in different partitions of RAM (memory blocks) as described in the table below.

When an 8-byte payload is selected:

- Block R0 allocates MB0 to MB31.
- Block R1 allocates MB32 to MB63.
- Block R2 allocates MB64 to MB95.

When a payload larger than 8 bytes is selected, the maximum number of MBs in a block is limited as described below:

Table 443. Number of message buffers

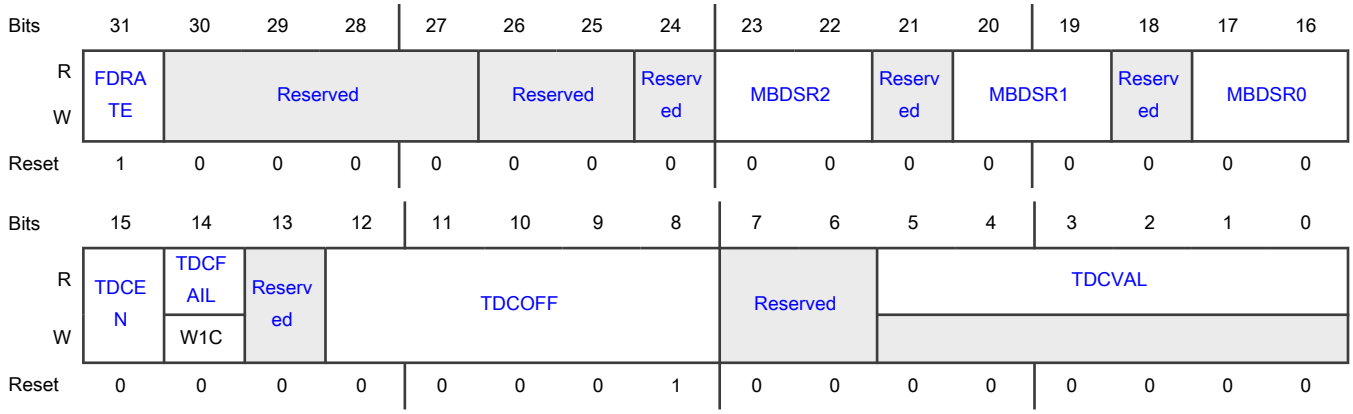
Payload size	Maximum number of message buffers per RAM block
8 bytes	32
16 bytes	21
32 bytes	12
64 bytes	7

NOTE

One memory block fits exactly 32 MBs with an 8-byte payload. For other possible payload sizes, empty memory may exist between the last MB in a block and the beginning of the next block. This empty memory corresponds to less than one MB, and must not be used.

The contents of this register are not affected by soft reset.

Diagram



Fields

Field	Function
31 FDRATE	Bit Rate Switch Enable This bit enables the effect of the Bit Rate Switch (BRS bit) during the data phase of Tx messages. The CPU can write this bit any time. However, its effect turns active only when the CAN bus is in Wait for Bus Idle, Bus Idle, or Bus Off state, or when the current frame under reception or transmission reaches the interframe space. By negating FDCTRL[FDRATE], the CPU can force all bits in CAN FD messages to be transmitted in nominal bit rate, no matter what the value is in the BRS bit of the Tx MBs.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - Transmit a frame in nominal rate. The BRS bit in the Tx MB has no effect.
1b - Transmit a frame with bit rate switching if the BRS bit in the Tx MB is recessive.</td></tr>
<tr><td>30-27
—</td><td>Reserved</td></tr>
<tr><td>26-25
—</td><td>Reserved</td></tr>
<tr><td>24
—</td><td>Reserved</td></tr>
<tr><td>23-22
MBDSR2</td><td>Message Buffer Data Size for Region 2
This two-bit field selects the data size (8, 16, 32, or 64 bytes) for region R2 of message buffers allocated in RAM.
It can be written in Freeze mode only.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>CAN_0</td><td>FDCTRL</td><td>—</td></tr>
<tr><td>CAN_1</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_2</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_3</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_4</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_5</td><td>—</td><td>FDCTRL</td></tr>
</table>
00b - Selects 8 bytes per message buffer.
01b - Selects 16 bytes per message buffer.
10b - Selects 32 bytes per message buffer.
11b - Selects 64 bytes per message buffer.</td></tr>
<tr><td>21
—</td><td>Reserved</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>20-19
MBDSR1</td><td>Message Buffer Data Size for Region 1
This two-bit field selects the data size (8, 16, 32, or 64 bytes) for region R1 of message buffers allocated in RAM.
It can be written in Freeze mode only.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>CAN_0</td><td>FDCTRL</td><td>—</td></tr>
<tr><td>CAN_1</td><td>FDCTRL</td><td>—</td></tr>
<tr><td>CAN_2</td><td>FDCTRL</td><td>—</td></tr>
<tr><td>CAN_3</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_4</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_5</td><td>—</td><td>FDCTRL</td></tr>
</table>
00b - Selects 8 bytes per message buffer.
01b - Selects 16 bytes per message buffer.
10b - Selects 32 bytes per message buffer.
11b - Selects 64 bytes per message buffer.</td></tr>
<tr><td>18
—</td><td>Reserved</td></tr>
<tr><td>17-16
MBDSR0</td><td>Message Buffer Data Size for Region 0
This two-bit field selects the data size (8, 16, 32, or 64 bytes) for region R0 of message buffers allocated in RAM.
It can be written in Freeze mode only.
00b - Selects 8 bytes per message buffer.
01b - Selects 16 bytes per message buffer.
10b - Selects 32 bytes per message buffer.
11b - Selects 64 bytes per message buffer.</td></tr>
<tr><td>15
TDCEN</td><td>Transceiver Delay Compensation Enable
This bit can be used to enable and disable the TDC feature. It can be written in Freeze mode only.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE See Transmitter delay compensation in the CAN Protocol standard (ISO 11898-1) for details. **NOTE** TDC must be disabled when Loop Back mode is enabled (see CTRL1[LPB]). **NOTE** If CTRL2[BTE] is set, TDCEN is read as zero and a write operation has no effect. 0b - TDC is disabled 1b - TDC is enabled
14 TDCFAIL	Transceiver Delay Compensation Fail This bit indicates when the Transceiver Delay Compensation (TDC) mechanism is out of range, unable to compensate the transceiver's loop delay and successfully compare the delayed received bits to the transmitted ones (see Transceiver delay compensation). TDCFAIL sets in the first time FlexCAN detects the out of range condition. The CPU needs to write one to clear it. **NOTE** If CTRL2[BTE] is set, TDCFAIL is read as zero and a write operation has no effect. 0b - Measured loop delay is in range. 1b - Measured loop delay is out of range.
13 —	Reserved
12-8 TDCOFF	Transceiver Delay Compensation Offset This bit field contains the offset value to be added to the measured transceiver's loop delay in order to define the position of the delayed comparison point when bit rate switching is active. See Transceiver delay compensation for more details on how the loop delay measurement is performed. TDCOFF can be written in Freeze mode only. Its value can be defined in Protocol Engine (PE) Clock periods (CANCLK, see Protocol timing for more details), and must be selected to be smaller than the CAN bit duration in the data bit rate for proper operation. **NOTE** If CTRL2[BTE] is set, TDCOFF is read as zero and a write operation has no effect. **NOTE** It is not recommended to have TDCOFF equal to zero.
7-6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-0 TDCVAL	Transceiver Delay Compensation Value This register contains the value of the transceiver loop delay measured from the transmitted EDL to R0 transition edge to the respective received one added to the TDCOFF value specified in the FDCTRL register. This value is an integer multiple of the Protocol Engine (PE) Clock period (CANCLK). **NOTE** If CTRL2[BTE] is set, TDCVAL is read as zero. See Protocol timing for more details on how the loop delay measurement is performed.

70.6.2.36 CAN FD Bit Timing Register (FDCBT)

Offset

Register	Offset
FDCBT	C04h

Function

This register stores the CAN bit timing variables used in the data phase of CAN FD messages when the FDCTRL[FDRATE] is set, compatible with CAN FD specification. FPRESDIV, FPROPSEG, FPSEG1, FPSEG2, and FRJW are used to define the time quantum duration, the number of time quanta per CAN bit, and the sample point position for the data bit rate portion of a CAN FD message with the BRS bit set.

The contents of this register are not affected by soft reset.

NOTE

The sum of the Fast Propagation Segment (FPROPSEG) and Fast Phase Segment 1 (FPSEG1) must be at least two time quanta.

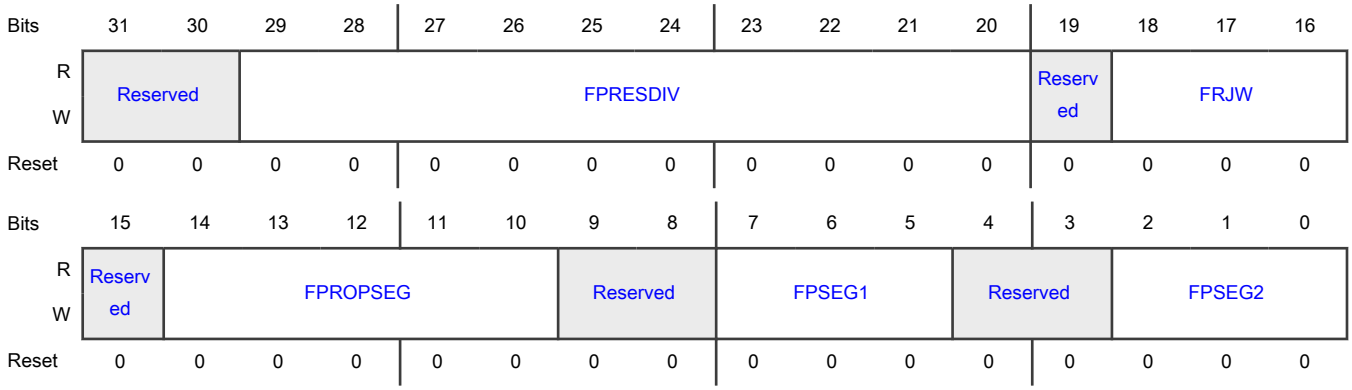
NOTE

The user must ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1).

NOTE

If CTRL2[BTE] is set, the FDCBT register is read as zero and a write operation has no effect.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-20 FPRESDIV	Fast Prescaler Division Factor This 10-bit field defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency in the data bit rate portion of a CAN FD message with the BRS bit set. The Sclock period defines the time quantum of the CAN FD protocol for the data bit rate. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Sclock frequency = PE clock frequency / (FPRESDIV + 1). **NOTE** To minimize errors when processing FD frames, use the same value for FPRESDIV and PRESDIV (in CBT or CTRL1). For more details see the first NOTE in section CAN FD frames.
19 —	Reserved
18-16 FRJW	Fast Resync Jump Width This 3-bit field defines the maximum number of time quanta that a bit time can be changed by one resynchronization in the data bit rate portion of a CAN FD message with the BRS bit set. One time quantum is equal to the Sclock period. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Resync Jump Width = FSJW + 1.
15 —	Reserved
14-10	Fast Propagation Segment

Table continues on the next page...

Table continued from the previous page...

Field	Function
FPROPSEG	This 5-bit field defines the length of the propagation segment in the bit time in the data bit rate portion of a CAN FD message with the BRS bit set. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Propagation Segment Time = FPROPSEG × Time-Quanta. Time-Quantum = one Sclock period.
9-8 —	Reserved
7-5 FPSEG1	Fast Phase Segment 1 This 3-bit field defines the length of phase segment 1 in the bit time in the data bit rate portion of a CAN FD message with the BRS bit set. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Segment 1 = (FPSEG1 + 1) × Time-Quanta. Time-Quantum = one Sclock period.
4-3 —	Reserved
2-0 FPSEG2	Fast Phase Segment 2 This 3-bit field defines the length of phase segment 2 in the data bit rate portion of a CAN FD message with the BRS bit set. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Segment 2 = (FPSEG2 + 1) × Time-Quanta. Time-Quantum = one Sclock period.

70.6.2.37 CAN FD CRC Register (FDCRC)

Offset

Register	Offset
FDCRC	C08h

Function

This register provides information about the CRC of transmitted messages.

FlexCAN uses different CRC polynomials for different frame formats, as shown below.

The CRC_15 polynomial is used for all frames in CAN format. The CRC_17 polynomial is used for frames in CAN FD format with a DATA FIELD up to sixteen bytes. The CRC_21 polynomial is used for frames in CAN FD format with a DATA FIELD longer than sixteen bytes. Each polynomial shown below results in a Hamming distance of 6. This register is updated at the same time the Tx Interrupt flag is asserted.

$$CRC_15 = C599h: \ (x^{15} + x^{14} + x^{10} + x^8 + x^7 + x^4 + x^3 + 1)$$

$$CRC_17 = 3685Bh: \ (x^{17} + x^{16} + x^{14} + x^{13} + x^{11} + x^6 + x^4 + x^3 + x^1 + 1)$$

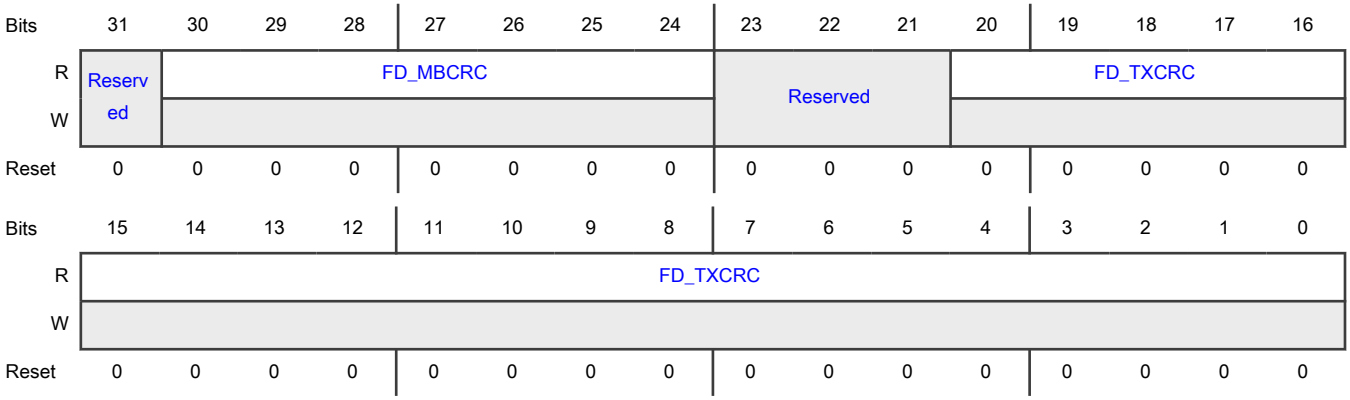
$$CRC_21 = 302899h: \ (x^{21} + x^{20} + x^{13} + x^{11} + x^7 + x^4 + x^3 + 1)$$

Equation 28. CRC polynomial used on CAN frame

NOTE

See CRC sequence calculation in the CAN Protocol standard (ISO 11898-1) for details.

Diagram



Fields

Field	Function
31 —	Reserved
30-24 FD_MBCRC	CRC Mailbox Number for FD_TXCRC This field indicates the number of the mailbox corresponding to the value in the FD_TXCRC field, for both FD and non-FD frames. It reports the same information as in CRCR[MBCRC].
23-21 —	Reserved
20-0 FD_TXCRC	Extended Transmitted CRC value This 21-bit field contains the CRC value calculated over the most recent transmitted message. Different CRC polynomials are used for different frame formats. A 15-bit polynomial, CRC_15, is used for all frames in CAN format. The second 17-bit polynomial, CRC_17, is used for frames in CAN FD format with a data field up to sixteen bytes long. The third 21-bit polynomial, CRC_21, is used for frames in CAN FD format with a data field longer than sixteen bytes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For CRC_15 and CRC_17, the 6 most significant bits and the 4 most significant bits are reported as zeros, respectively. For CRC_15, this register has the same content as CRC register.

70.6.2.38 Enhanced Rx FIFO Control Register (ERFCR)

Offset

Register	Offset
ERFCR	C0Ch

Function

This register defines the Enhanced Rx FIFO configuration.

This register can be written only in Freeze mode.

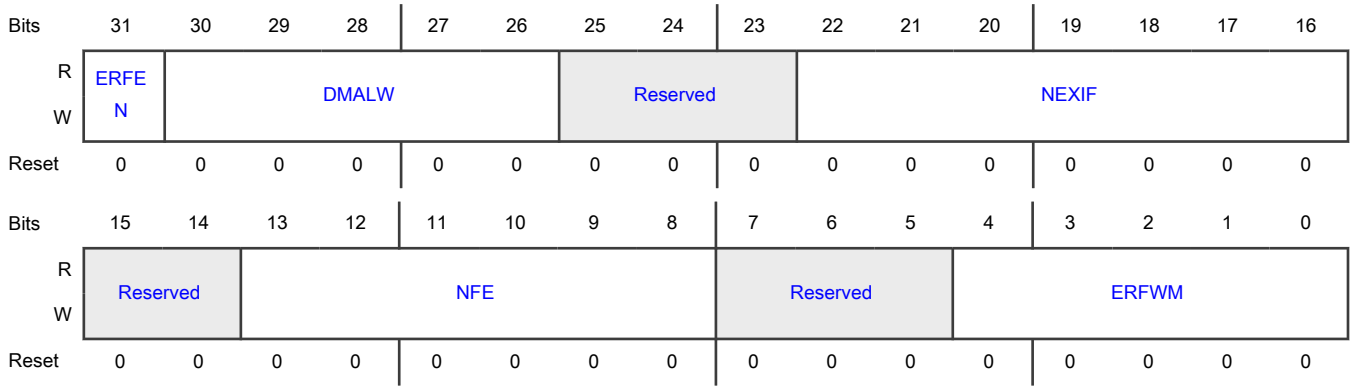
All the contents of this register are affected by soft reset.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
CAN_0	ERFCR	—
CAN_1	—	ERFCR
CAN_2	—	ERFCR
CAN_3	—	ERFCR
CAN_4	—	ERFCR
CAN_5	—	ERFCR

Diagram



Fields

Field	Function
31 ERFEN	Enhanced Rx FIFO enable This field enables the Enhanced Rx FIFO. **NOTE** ERFEN must not be set if MCR[RFEN] is set. 0b - Enhanced Rx FIFO is disabled 1b - Enhanced Rx FIFO is enabled
30-26 DMALW	DMA Last Word This field defines the last DMA address for each Enhanced RX FIFO element. The table shows the number of elements and the last address for each Enhanced Rx FIFO element according to the value of DMALW.

DMALW	Number of 32-bit words transferred	Last FIFO address
0	1	2000h
1	2	2004h
2	3	2008h
3	4	200Ch
4	5	2010h
5	6	2014h
6	7	2018h
7	8	201Ch
8	9	2020h
9	10	2024h
10	11	2028h

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>DMALW</th><th>Number of 32-bit words transferred</th><th>Last FIFO address</th></tr>
<tr><td>11</td><td>12</td><td>202Ch</td></tr>
<tr><td>12</td><td>13</td><td>2030h</td></tr>
<tr><td>13</td><td>14</td><td>2034h</td></tr>
<tr><td>14</td><td>15</td><td>2038h</td></tr>
<tr><td>15</td><td>16</td><td>203Ch</td></tr>
<tr><td>16</td><td>17</td><td>2040h</td></tr>
<tr><td>17</td><td>18</td><td>2044h</td></tr>
<tr><td>18</td><td>19</td><td>2048h</td></tr>
<tr><td>19</td><td>20</td><td>204Ch</td></tr>
</table>
NOTE
Undefined DMALW values in the table are reserved and must not be used.
</td></tr>
<tr><td>25-23
—</td><td>Reserved</td></tr>
<tr><td>22-16
NEXIF</td><td>Number of Extended ID Filter Elements
This field defines the number of extended ID filter elements used during the Enhanced Rx FIFO matching process.
NEXIF must be less than or equal to NFE + 1.
The number of standard ID filter elements is 2 × (NFE – NEXIF + 1).
The table shows the number of extended ID filters and standard ID filters available for Enhanced Rx FIFO if all the filter elements are used.
<table>
<tr><th>NEXIF</th><th>NFE</th><th>Number of Extended ID filter elements</th><th>Number of Standard ID filter elements</th></tr>
<tr><td>0</td><td>63</td><td>0</td><td>128</td></tr>
<tr><td>1</td><td>63</td><td>1</td><td>126</td></tr>
<tr><td>2</td><td>63</td><td>2</td><td>124</td></tr>
<tr><td>3</td><td>63</td><td>3</td><td>122</td></tr>
<tr><td>4</td><td>63</td><td>4</td><td>120</td></tr>
<tr><td>5</td><td>63</td><td>5</td><td>118</td></tr>
<tr><td>6</td><td>63</td><td>6</td><td>116</td></tr>
<tr><td>7</td><td>63</td><td>7</td><td>114</td></tr>
</table>
</td></tr>
</table>

Field	Function

NEXIF	NFE	Number of Extended ID filter elements	Number of Standard ID filter elements
8	63	8	112
9	63	9	110
10	63	10	108
11	63	11	106
12	63	12	104
13	63	13	102
14	63	14	100
15	63	15	98
16	63	16	96
17	63	17	94
18	63	18	92
19	63	19	90
20	63	20	88
21	63	21	86
22	63	22	84
23	63	23	82
24	63	24	80
25	63	25	78
26	63	26	76
27	63	27	74
28	63	28	72
29	63	29	70
30	63	30	68
31	63	31	66
32	63	32	64
33	63	33	62
34	63	34	60
35	63	35	58
36	63	36	56

Table continued from the previous page...

Field	Function			
	NEXIF	NFE	Number of Extended ID filter elements	Number of Standard ID filter elements
	37	63	37	54
	38	63	38	52
	39	63	39	50
	40	63	40	48
	41	63	41	46
	42	63	42	44
	43	63	43	42
	44	63	44	40
	45	63	45	38
	46	63	46	36
	47	63	47	34
	48	63	48	32
	49	63	49	30
	50	63	50	28
	51	63	51	26
	52	63	52	24
	53	63	53	22
	54	63	54	20
	55	63	55	18
	56	63	56	16
	57	63	57	14
	58	63	58	12
	59	63	59	10
	60	63	60	8
	61	63	61	6
	62	63	62	4
	63	63	63	2
	64	63	64	0

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>15-14
—</td><td>Reserved</td></tr>
<tr><td>13-8
NFE</td><td>Number of Enhanced Rx FIFO Filter Elements
This field defines the total number of filter elements used during the enhanced RX FIFO matching process according to the table.
<table>
<tr><th>NFE</th><th>Maximum number of Extended ID filter elements (NEXIF = NFE + 1)</th><th>Maximum number of standard ID filter elements (NEXIF = 0)</th></tr>
<tr><td>0</td><td>1</td><td>2</td></tr>
<tr><td>1</td><td>2</td><td>4</td></tr>
<tr><td>2</td><td>3</td><td>6</td></tr>
<tr><td>3</td><td>4</td><td>8</td></tr>
<tr><td>4</td><td>5</td><td>10</td></tr>
<tr><td>5</td><td>6</td><td>12</td></tr>
<tr><td>6</td><td>7</td><td>14</td></tr>
<tr><td>7</td><td>8</td><td>16</td></tr>
<tr><td>8</td><td>9</td><td>18</td></tr>
<tr><td>9</td><td>10</td><td>20</td></tr>
<tr><td>10</td><td>11</td><td>22</td></tr>
<tr><td>11</td><td>12</td><td>24</td></tr>
<tr><td>12</td><td>13</td><td>26</td></tr>
<tr><td>13</td><td>14</td><td>28</td></tr>
<tr><td>14</td><td>15</td><td>30</td></tr>
<tr><td>15</td><td>16</td><td>32</td></tr>
<tr><td>16</td><td>17</td><td>34</td></tr>
<tr><td>17</td><td>18</td><td>36</td></tr>
<tr><td>18</td><td>19</td><td>38</td></tr>
<tr><td>19</td><td>20</td><td>40</td></tr>
<tr><td>20</td><td>21</td><td>42</td></tr>
<tr><td>21</td><td>22</td><td>44</td></tr>
<tr><td>22</td><td>23</td><td>46</td></tr>
<tr><td>23</td><td>24</td><td>48</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function		
	NFE	Maximum number of Extended ID filter elements (NEXIF = NFE + 1)	Maximum number of standard ID filter elements (NEXIF = 0)
	24	25	50
	25	26	52
	26	27	54
	27	28	56
	28	29	58
	29	30	60
	30	31	62
	31	32	64
	32	33	66
	33	34	68
	34	35	70
	35	36	72
	36	37	74
	37	38	76
	38	39	78
	39	40	80
	40	41	82
	41	42	84
	42	43	86
	43	44	88
	44	45	90
	45	46	92
	46	47	94
	47	48	96
	48	49	98
	49	50	100
	50	51	102
	51	52	104
	52	53	106

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>NFE</th><th>Maximum number of Extended ID filter elements (NEXIF = NFE + 1)</th><th>Maximum number of standard ID filter elements (NEXIF = 0)</th></tr><tr><td>53</td><td>54</td><td>108</td></tr><tr><td>54</td><td>55</td><td>110</td></tr><tr><td>55</td><td>56</td><td>112</td></tr><tr><td>56</td><td>57</td><td>114</td></tr><tr><td>57</td><td>58</td><td>116</td></tr><tr><td>58</td><td>59</td><td>118</td></tr><tr><td>59</td><td>60</td><td>120</td></tr><tr><td>60</td><td>61</td><td>122</td></tr><tr><td>61</td><td>62</td><td>124</td></tr><tr><td>62</td><td>63</td><td>126</td></tr><tr><td>63</td><td>64</td><td>128</td></tr></table>
7-5 —	Reserved
4-0 ERFWM	Enhanced Rx FIFO Watermark This field defines the minimum number of CAN messages stored in the Enhanced RX FIFO. When that number is reached, then ERFSR[ERFWMI] is set. Minimum number of CAN messages = ERFWM + 1. **NOTE** If MCR[DMA] is set, ERFWM field should be configures as 0x0.

70.6.2.39 Enhanced Rx FIFO Interrupt Enable Register (ERFIER)

Offset

Register	Offset
ERFIER	C10h

Function

This register contains the interrupt enables for the Enhanced Rx FIFO.

This register is affected by soft reset.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
CAN_0	ERFIER	—
CAN_1	—	ERFIER
CAN_2	—	ERFIER
CAN_3	—	ERFIER
CAN_4	—	ERFIER
CAN_5	—	ERFIER

Diagram

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

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

Fields

Field	Function
31 ERFUFWIE	Enhanced Rx FIFO Underflow Interrupt Enable This field enables interrupt for ERFSR[ERFUFW]. 0b - Enhanced Rx FIFO Underflow interrupt is disabled 1b - Enhanced Rx FIFO Underflow interrupt is enabled
30 ERFOVFIE	Enhanced Rx FIFO Overflow Interrupt Enable This field enables interrupt for ERFSR[ERFOVF]. 0b - Enhanced Rx FIFO Overflow is disabled 1b - Enhanced Rx FIFO Overflow is enabled
29 ERFWMIIE	Enhanced Rx FIFO Watermark Indication Interrupt Enable This field enables interrupt for ERFSR[ERFWMI].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enhanced Rx FIFO Watermark interrupt is disabled 1b - Enhanced Rx FIFO Watermark interrupt is enabled
28 ERFDAIE	Enhanced Rx FIFO Data Available Interrupt Enable This field enables interrupt for ERFSR[ERFDA]. 0b - Enhanced Rx FIFO Data Available interrupt is disabled 1b - Enhanced Rx FIFO Data Available interrupt is enabled
27-16 —	Reserved
15 —	Reserved
14-0 —	Reserved

70.6.2.40 Enhanced Rx FIFO Status Register (ERFSR)

Offset

Register	Offset
ERFSR	C14h

Function

This register contains the status bits of the Enhanced RX FIFO including error indications and a clear FIFO bit.

This register is affected by soft reset.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
CAN_0	ERFSR	—
CAN_1	—	ERFSR
CAN_2	—	ERFSR
CAN_3	—	ERFSR

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
CAN_4	—	ERFSR
CAN_5	—	ERFSR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ERFUFW	ERFOVF	ERFWMI	ERFDA	0	0									ERFE	ERFF
W	W1C	W1C	W1C	W1C	ERFCLR											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31 ERFUFW	Enhanced Rx FIFO Underflow This field indicates that an underflow condition occurred in the enhanced Rx FIFO. This field generates an interrupt if ERFIER[ERFUFWIE] is set. 0b - No such occurrence 1b - Enhanced Rx FIFO underflow
30 ERFOVF	Enhanced Rx FIFO Overflow This field indicates that an overflow condition occurred in the Enhanced Rx FIFO. This field generates an interrupt if ERFIER[ERFOVFIE] is set. 0b - No such occurrence 1b - Enhanced Rx FIFO overflow
29 ERFWMI	Enhanced Rx FIFO Watermark Indication This field is set by the hardware if the number of messages available in the Enhanced Rx FIFO is greater than the watermark defined in ERFCR[ERFWM]. This field generates an interrupt if ERFIER[ERFWMIIE] is set. 0b - No such occurrence

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - The number of messages in FIFO is greater than the watermark
28 ERFDA	Enhanced Rx FIFO Data Available This field is set by the hardware when there is at least one message stored in the ERX FIFO. This field generates an interrupt if ERFIER[ERFDAIE] is set. 0b - No such occurrence 1b - There is at least one message stored in Enhanced Rx FIFO
27 ERFCLR	Enhanced Rx FIFO Clear Writing one to this field during Freeze mode clears Enhanced Rx FIFO content. Writing out of Freeze mode or writing zero to this field has no effect. 0b - No effect 1b - Clear Enhanced Rx FIFO content
26-18 —	Reserved
17 ERFE	Enhanced Rx FIFO empty This field indicates that Enhanced Rx FIFO is empty. 0b - Enhanced Rx FIFO is not empty 1b - Enhanced Rx FIFO is empty
16 ERFF	Enhanced Rx FIFO full This field indicates that Enhanced Rx FIFO is full. 0b - Enhanced Rx FIFO is not full 1b - Enhanced Rx FIFO is full
15 —	Reserved
14-6 —	Reserved
5-0 ERFEL	Enhanced Rx FIFO Elements This field indicates the number of CAN messages stored in the Enhanced Rx FIFO.

70.6.2.41 High Resolution Time Stamp (HR_TIME_STAMP0 - HR_TIME_STAMP95)

Offset

For n = 0 to 95:

Register	Offset
HR_TIME_STAMPn	C30h + (n × 4h)

Function

The High Resolution Time Stamp registers are used for storing a copy of a 32-bit timer at the start or end of a CAN frame.

HRST0 stores the 32-bit time stamp associated with MB0, HRSTS1 stores the the 32-bit time stamp associated with MB1, and so on.

These registers are not affected by reset.

NOTE

These registers must not be written out of Freeze mode.

Table 444. High Resolution Time Stamp register operation

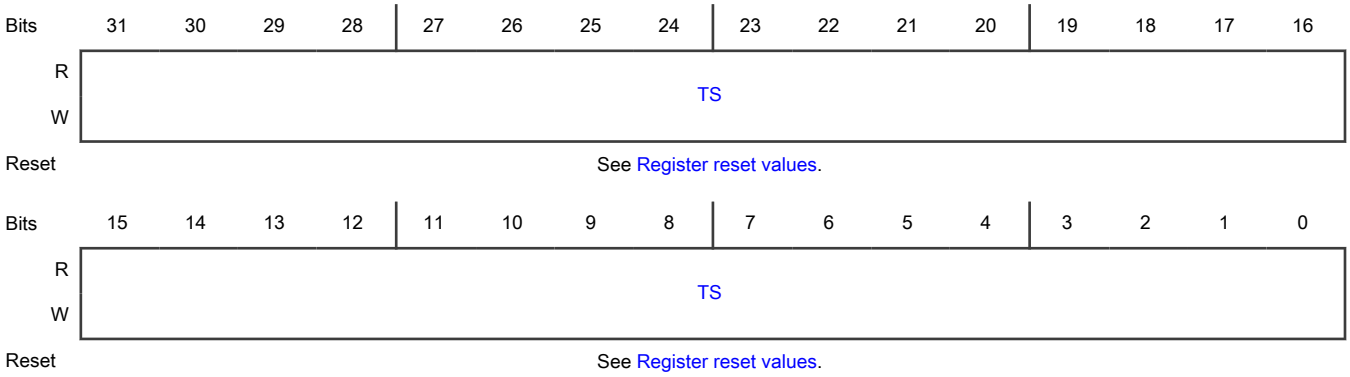
TSTAMPCAP	Captured time base	Capture point
b00	None	None
b01	32 bits of high-resolution on-chip timer	Seventh bit of the end of frame field for transmission and sixth bit of the end of frame field for reception.
b10	32 bits of high-resolution on-chip timer	Start of frame.
b11	32 bits of high-resolution on-chip timer	Start of frame for classical CAN frame format and res bit for CAN FD frame format.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
CAN_0	HR_TIME_STAMP0–HR_TIME_STAMP95	—
CAN_1	HR_TIME_STAMP0–HR_TIME_STAMP63	HR_TIME_STAMP64–HR_TIME_STAMP95
CAN_2	HR_TIME_STAMP0–HR_TIME_STAMP63	HR_TIME_STAMP64–HR_TIME_STAMP95
CAN_3	HR_TIME_STAMP0–HR_TIME_STAMP31	HR_TIME_STAMP32–HR_TIME_STAMP95
CAN_4	HR_TIME_STAMP0–HR_TIME_STAMP31	HR_TIME_STAMP32–HR_TIME_STAMP95
CAN_5	HR_TIME_STAMP0–HR_TIME_STAMP31	HR_TIME_STAMP32–HR_TIME_STAMP95

Diagram



Register reset values

Register	Reset value
HR_TIME_STAMP0–HR_TIME_STAMP31	CAN_0–CAN_5: undefined
HR_TIME_STAMP32–HR_TIME_STAMP63	CAN_0–CAN_2: undefined CAN_3–CAN_5: Register not supported
HR_TIME_STAMP64–HR_TIME_STAMP95	CAN_0: undefined CAN_1–CAN_5: Register not supported

Fields

Field	Function
31-0 TS	High Resolution Time Stamp HR_TIME_STAMP register which captures the copy of timestamp of corresponding MB always captures 32-bit timer value.

70.6.2.42 Enhanced Rx FIFO Filter Element (ERFFEL0 - ERFFEL127)

Offset

For n = 0 to 127:

Register	Offset
ERFFELn	3000h + (n × 4h)

Function

These registers are used for storing the filter elements of the Enhanced Rx FIFO.

For standard ID filtering, each ERFFEL register stores one filter element. For extended ID filtering, each pair of ERFFEL registers stores one filter element.

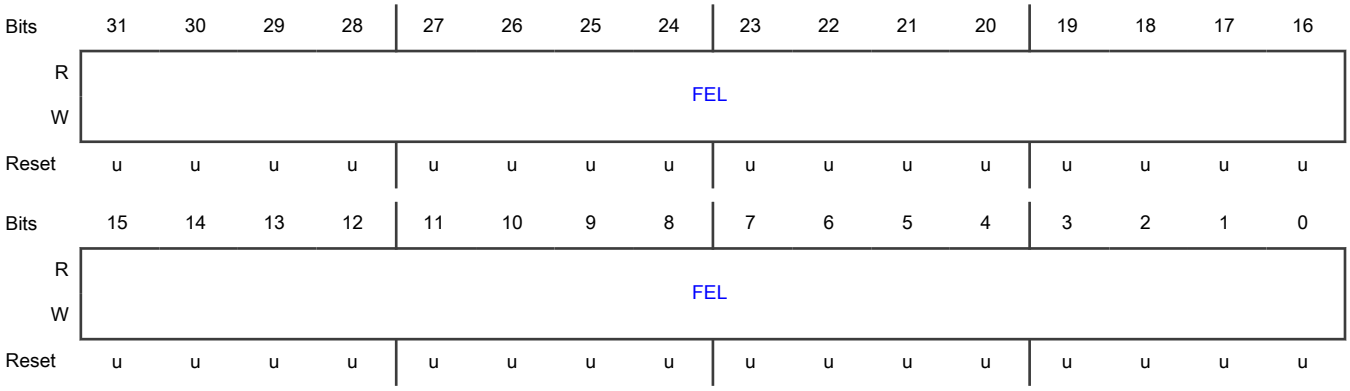
ERFFEL registers can be written only in Freeze mode; otherwise, they are blocked by hardware. These registers are not affected by reset. They are located in RAM and must be explicitly initialized prior to any reception.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
CAN_0	ERFFEL0–ERFFEL127	—
CAN_1	—	ERFFEL0–ERFFEL127
CAN_2	—	ERFFEL0–ERFFEL127
CAN_3	—	ERFFEL0–ERFFEL127
CAN_4	—	ERFFEL0–ERFFEL127
CAN_5	—	ERFFEL0–ERFFEL127

Diagram



Fields

Field	Function
31-0 FEL	Filter Element Bits Each filter element is used during the match process and, if the matching criteria are met, a message is stored in the Enhanced Rx FIFO.

70.6.3 Message buffer structure

The message buffer structure used by the FlexCAN module is represented in the following figure. Both extended (29-bit identifier) and standard (11-bit identifier) frames used in the CAN specification (Version 2.0 Part B) are represented. Each individual MB is formed by 16, 24, 40, or 72 bytes, depending on the quantity of data bytes allocated for the message payload: 8, 16, 32, or 64 data bytes, respectively.

The memory area 80h–67Fh is used by the mailboxes. When CAN FD is enabled, the exact address for each MB depends on the size of its payload. See FlexCAN memory partition for CAN FD for more detailed information.

Table 445. Message buffer structure — example with 64-byte payload

	31	30	29	28	27	24	23	22	21	20	19	18	17	16	15	8	7	0
0h	EDL	BRS	ESI		CODE			SRR	IDE	RTR	DLC				TIME STAMP			
4h	PRIO				ID (standard/extended)										ID (extended)			
8h	Data byte 0						Data byte 1								Data byte 2		Data byte 3	
Ch	Data byte 4						Data byte 5								Data byte 6		Data byte 7	
10h	Data byte 8						Data byte 9								Data byte 10		Data byte 11	
14h	Data byte 12						Data byte 13								Data byte 14		Data byte 15	
18h	Data byte 16						Data byte 17								Data byte 18		Data byte 19	
1Ch	Data byte 20						Data byte 21								Data byte 22		Data byte 23	
20h	Data byte 24						Data byte 25								Data byte 26		Data byte 27	
24h	Data byte 28						Data byte 29								Data byte 30		Data byte 31	
28h	Data byte 32						Data byte 33								Data byte 34		Data byte 35	
2Ch	Data byte 36						Data byte 37								Data byte 38		Data byte 39	
30h	Data byte 40						Data byte 41								Data byte 42		Data byte 43	
34h	Data byte 44						Data byte 45								Data byte 46		Data byte 47	
38h	Data byte 48						Data byte 49								Data byte 50		Data byte 51	
3Ch	Data byte 52						Data byte 53								Data byte 54		Data byte 55	
40h	Data byte 56						Data byte 57								Data byte 58		Data byte 59	
44h	Data byte 60						Data byte 61								Data byte 62		Data byte 63	

= Unimplemented or reserved

EDL — Extended Data Length

This bit distinguishes between CAN format and CAN FD format frames. The EDL bit must not be set for message buffers configured to RANSWER with code field 1010b (see table below).

BRS — Bit Rate Switch

This bit defines whether the bit rate is switched inside a CAN FD format frame.

ESI — Error State Indicator

This bit indicates if the transmitting node is error active or error passive.

CODE — Message Buffer Code

This 4-bit field can be accessed (read or write) by the CPU and by the FlexCAN module itself, as part of the message buffer matching and arbitration process. The encoding is shown in Table 446 and Table 447. See Functional description for additional information.

Table 446. Message buffer code for Rx buffers

CODE description	Rx code BEFORE receive new frame	SRV[1]	Rx code AFTER successful reception[2]	RRS[3]	Comment
0000b: INACTIVE — MB is not active.	INACTIVE	—	—	—	MB does not participate in the matching process.
0100b: EMPTY — MB is active and empty.	EMPTY	—	FULL	—	When a frame is received successfully (after the Move-in process), the CODE field is automatically updated to FULL.
0010b: FULL — MB is full.	FULL	Yes	FULL	—	The act of reading the C/S word followed by unlocking the MB (SRV) does not make the code return to EMPTY. It remains FULL. If a new frame is moved to the MB after the MB was serviced, the code still remains FULL. See Matching process for matching details related to FULL code.
		No	OVERRUN	—	If the MB is FULL and a new frame is moved to this MB before the CPU services it, the CODE field is automatically updated to OVERRUN. See Matching process for details about overrun behavior.

Table continues on the next page...

Table 446. Message buffer code for Rx buffers (continued)

CODE description	Rx code BEFORE receive new frame	SRV[1]	Rx code AFTER successful reception[2]	RRS[3]	Comment
0110b: OVERRUN — MB is being overwritten into a full buffer.	OVERRUN	Yes	FULL	—	If the CODE field indicates OVERRUN and CPU has serviced the MB, when a new frame is moved to the MB then the code returns to FULL.
		No	OVERRUN	—	If the CODE field already indicates OVERRUN, and another new frame must be moved, the MB will be overwritten again, and the code will remain OVERRUN. See Matching process for details about overrun behavior.
1010b: RANSWER[4] — A frame was configured to recognize a Remote Request frame and transmit a Response frame in return.[5]	RANSWER	—	TANSWER(1110b)	0	A Remote Answer was configured to recognize a remote request frame received. After that an MB is set to transmit a response frame. The code is automatically changed to TANSWER (1110b). See Matching process for details. If CTRL2[RRS] is negated, transmit a response frame whenever a remote request frame with the same ID is received.
		—	—	1	This code is ignored during

Table continues on the next page...

Table 446. Message buffer code for Rx buffers (continued)

CODE description	Rx code BEFORE receive new frame	SRV[1]	Rx code AFTER successful reception[2]	RRS[3]	Comment
					matching and arbitration process. See Matching process for details.
CODE[0]=1: BUSY — FlexCAN is updating the contents of the MB. The CPU must not access the MB.	BUSY[6]	—	FULL	—	Indicates that the MB is being updated. It will be negated automatically and does not interfere with the next CODE.
		—	OVERRUN	—	

1. SRV: Serviced MB. MB was read and unlocked by reading TIMER or other MB.
2. A frame is considered a successful reception after the frame to be moved to MB (move-in process). See Move-in for details.
3. Remote Request Stored bit, see "Control 2 register (CTRL2)" for details.
4. Code 1010b is not considered Tx and an MB with this code should not be aborted.
5. Code 1010b must be used in message buffers configured in CAN FD format, having the EDL bit set.
6. Note that for Tx MBs, the BUSY bit should be ignored upon read, except when AEN bit is set in the MCR register. If this bit is asserted, the corresponding MB does not participate in the matching process.

Table 447. Message buffer code for Tx buffers

CODE Description	Tx Code BEFORE tx frame	MB RTR	Tx Code AFTER successful transmission	Comment
1000b: INACTIVE — MB is not active	INACTIVE	—	—	MB does not participate in arbitration process.
1001b: ABORT — MB is aborted	ABORT	—	—	MB does not participate in arbitration process.
1100b: DATA — MB is a Tx data frame (MB RTR must be 0)	DATA	0	INACTIVE	Transmit data frame unconditionally once. After transmission, the MB automatically returns to the INACTIVE state.
1100b: REMOTE — MB is a Tx Remote Request frame (MB RTR must be 1)	REMOTE	1	EMPTY	Transmit remote request frame unconditionally once. After transmission, the MB automatically becomes an Rx Empty MB with the same ID.

Table continues on the next page...

Table 447. Message buffer code for Tx buffers (continued)

CODE Description	Tx Code BEFORE tx frame	MB RTR	Tx Code AFTER successful transmission	Comment
1110b: TANSWER — MB is a Tx Response frame from an incoming Remote Request frame	TANSWER	—	RANSWER	This is an intermediate code that is automatically written to the MB by the CHI as a result of a match to a remote request frame. The remote response frame will be transmitted unconditionally once, and then the code will automatically return to RANSWER (1010b). The CPU can also write this code with the same effect. The remote response frame can be either a data frame or another remote request frame depending on the RTR bit value. See Matching process and Arbitration process for details.

SRR — Substitute Remote Request

Fixed recessive bit, used only in extended format. It must be set to one by the user for transmission (Tx Buffers) and will be stored with the value received on the CAN bus for Rx receiving buffers. It can be received as either recessive or dominant. If FlexCAN receives this bit as dominant, then it is interpreted as an arbitration loss.

1 = Recessive value is compulsory for transmission in extended format frames

0 = Dominant is not a valid value for transmission in extended format frames

IDE — ID Extended Bit

This field identifies whether the frame format is standard or extended.

1 = Frame format is extended

0 = Frame format is standard

RTR — Remote Transmission Request

This bit affects the behavior of remote frames and is part of the reception filter. See Table 446, Table 447, and the description of the RRS field in Control 2 register (CTRL2) for additional details.

If FlexCAN transmits this bit as '1' (recessive) and receives it as '0' (dominant), it is interpreted as an arbitration loss. If this bit is transmitted as '0' (dominant), then if it is received as '1' (recessive), the FlexCAN module treats it as a bit error. If the value received matches the value transmitted, it is considered a successful bit transmission.

1 = Indicates the current MB may have a remote request frame to be transmitted if MB is Tx. If the MB is Rx then incoming remote request frames may be stored.

0 = Indicates the current MB has a data frame to be transmitted. In Rx MB it may be considered in matching processes.

NOTE

When configuring CAN FD frames, the RTR bit must be negated.

DLC — Length of Data in Bytes

This 4-bit field is the length (in bytes) of the Rx or Tx data, which is located in offset 8h–Fh of the MB space (see Table 445). In reception, this field is written by the FlexCAN module, copied from the DLC (Data Length Code) field of the received frame. In transmission, this field is written by the CPU and corresponds to the DLC field value of the frame to be transmitted. When RTR = 1, the frame to be transmitted is a remote frame and does not include the data field, regardless of the DLC field (see Table 449).

TIME STAMP — Free-Running Counter Time Stamp

This 16-bit field is a copy of the Free-Running Timer, captured for Tx and Rx frames at the time when the beginning of the Identifier field appears on the CAN bus.

Table 448. Time stamp operation

TSTAMPCAP	MBTSBASE	TIMER_SOURCE	Captured time base	Capture point
b00	bxx	0	CAN_TIMER incremented by CAN bit clock	Second bit of identifier field
b00	bxx	1	CAN_TIMER incremented by on-chip timer clock	Second bit of identifier field
bxx	b00	0	CAN_TIMER incremented by CAN bit clock	Second bit of identifier field
bxx	b00	1	CAN_TIMER incremented by on-chip timer clock	Second bit of identifier field
b01	b01	x	Lower 16 bits of high-resolution on-chip timer	Seventh bit of the end of frame field for transmission and sixth bit of the end of frame field for reception
b01	b10	x	Upper 16 bits of high-resolution on-chip timer	Seventh bit of the end of frame field for transmission and sixth bit of the end of frame field for reception
b10	b01	x	Lower 16 bits of high-resolution on-chip timer	Start of frame
b10	b10	x	Upper 16 bits of high-resolution on-chip timer	Start of frame
b11	b01	x	Lower 16 bits of high-resolution on-chip timer	Start of frame for classical CAN frame format and res bit for CAN FD frame format
b11	b10	x	Upper 16 bits of high-resolution on-chip timer	Start of frame for classical CAN frame

Table continues on the next page...

Table 448. Time stamp operation (continued)

TSTAMPCAP	MBTSBASE	TIMER_SOURCE	Captured time base	Capture point
				format and res bit for CAN FD frame format

PRIO — Local priority

This 3-bit field is used only when MCR[LPRIO_EN] is set, and it only makes sense for Tx mailboxes. These bits are not transmitted. They are appended to the regular ID to define the transmission priority. See Arbitration process.

ID — Frame Identifier

In standard frame format, only the 11 most significant bits (28 to 18) are used for frame identification in both receive and transmit cases. The 18 least significant bits are ignored. In extended frame format, all bits are used for frame identification in both receive and transmit cases.

DATA BYTE 0 to 63 — Data Field

Up to sixty four bytes can be used for a data frame, depending on the size of payload selected for the message buffers.

For Rx frames, the data is stored as it is received from the CAN bus. DATA BYTE (*n*) is valid only if *n* is less than DLC as shown in the table below.

Table 449. DATA BYTEs validity

DLC	Valid DATA BYTEs
0	none
1	DATA BYTE 0
2	DATA BYTE 0 to 1
3	DATA BYTE 0 to 2
4	DATA BYTE 0 to 3
5	DATA BYTE 0 to 4
6	DATA BYTE 0 to 5
7	DATA BYTE 0 to 6
8	DATA BYTE 0 to 7
9	DATA BYTE 0 to 11
10	DATA BYTE 0 to 15
11	DATA BYTE 0 to 19
12	DATA BYTE 0 to 23
13	DATA BYTE 0 to 31
14	DATA BYTE 0 to 47
15	DATA BYTE 0 to 63

70.6.4 FlexCAN memory partition for CAN FD

When CAN FD is enabled, the FlexCAN RAM can be partitioned in blocks of 512 bytes. Each block can accommodate a number of message buffers which depends on the configuration provided by FDCTRL[MBDSRn] bit fields as shown in table below.

Table 450. RAM partition

RAM block	Number of MBs with 8 bytes (default range)	Size control bit field in FDCTRL register	Number of MBs of different sizes, per block
0	0 to 31	MBDSR0	MBDSR0=00, 32 MBs with 8 bytes payload MBDSR0=01, 21 MBs with 16 bytes payload MBDSR0=10, 12 MBs with 32 bytes payload MBDSR0=11, 7 MBs with 64 bytes payload
1	32 to 63	MBDSR1	MBDSR1=00, 32 MBs with 8 bytes payload MBDSR1=01, 21 MBs with 16 bytes payload MBDSR1=10, 12 MBs with 32 bytes payload MBDSR1=11, 7 MBs with 64 bytes payload
2	64 to 95	MBDSR2	MBDSR2=00, 32 MBs with 8 bytes payload MBDSR2=01, 21 MBs with 16 bytes payload MBDSR2=10, 12 MBs with 32 bytes payload MBDSR2=11, 7 MBs with 64 bytes payload

When payload sizes of 16, 32, or 64 bytes are configured in some or all RAM blocks, the total number of MBs and its respective number order may differ from the default configuration of 8 bytes. For example, suppose Block0 is configured to 8 bytes payload, Block1 to 16 bytes, Block2 to 32 bytes, Then the following table indicates how the message buffers will be arranged in RAM.

Table 451. RAM partition example

RAM block	Payload size	Number of MBs in the RAM block	Message buffer range
0	FDCTRL[MBDSR0]=00, 8 bytes payload	32	0 to 31
1	FDCTRL[MBDSR1]=01, 16 bytes payload	21	32 to 52
2	FDCTRL[MBDSR2]=10, 32 bytes payload	12	53 to 64

70.6.5 FlexCAN message buffer memory map

The FlexCAN memory buffers are allocated in memory according to the tables below.

Table 452. 8-byte message buffers

Address offset (hex)	MBDSR=b00 8-byte payload
0080	MB0
0090	MB1
00A0	MB2
00B0	MB3
00C0	MB4
00D0	MB5
00E0	MB6
00F0	MB7
0100	MB8
0110	MB9
0120	MB10
0130	MB11
0140	MB12
0150	MB13
0160	MB14
0170	MB15
0180	MB16
0190	MB17
01A0	MB18
01B0	MB19
01C0	MB20
01D0	MB21
01E0	MB22
01F0	MB23
0200	MB24
0210	MB25
0220	MB26
0230	MB27
0240	MB28

Table continues on the next page...

Table 452. 8-byte message buffers (continued)

Address offset (hex)	MBDSR=b00 8-byte payload
0250	MB29
0260	MB30
0270	MB31
0280	MB32
0290	MB33
02A0	MB34
02B0	MB35
02C0	MB36
02D0	MB37
02E0	MB38
02F0	MB39
0300	MB40
0310	MB41
0320	MB42
0330	MB43
0340	MB44
0350	MB45
0360	MB46
0370	MB47
0380	MB48
0390	MB49
03A0	MB50
03B0	MB51
03C0	MB52
03D0	MB53
03E0	MB54
03F0	MB55
0400	MB56
0410	MB57
0420	MB58
0430	MB59

Table continues on the next page...

Table 452. 8-byte message buffers (continued)

Address offset (hex)	MBDSR=b00 8-byte payload
0440	MB60
0450	MB61
0460	MB62
0470	MB63
0480	MB64
0490	MB65
04A0	MB66
04B0	MB67
04C0	MB68
04D0	MB69
04E0	MB70
04F0	MB71
0500	MB72
0510	MB73
0520	MB74
0530	MB75
0540	MB76
0550	MB77
0560	MB78
0570	MB79
0580	MB80
0590	MB81
05A0	MB82
05B0	MB83
05C0	MB84
05D0	MB85
05E0	MB86
05F0	MB87
0600	MB88
0610	MB89
0620	MB90

Table continues on the next page...

Table 452. 8-byte message buffers (continued)

Address offset (hex)	MBDSR=b00 8-byte payload
0630	MB91
0640	MB92
0650	MB93
0660	MB94
0670	MB95

Table 453. 16-byte message buffers

Address offset (hex)	MBDSR=b01 16-byte payload
0080	MB0
0098	MB1
00B0	MB2
00C8	MB3
00E0	MB4
00F8	MB5
0110	MB6
0128	MB7
0140	MB8
0158	MB9
0170	MB10
0188	MB11
01A0	MB12
01B8	MB13
01D0	MB14
01E8	MB15
0200	MB16
0218	MB17
0230	MB18
0248	MB19
0260	MB20
0280	MB21

Table continues on the next page...

Table 453. 16-byte message buffers (continued)

Address offset (hex)	MBDSR=b01 16-byte payload
0298	MB22
02B0	MB23
02C8	MB24
02E0	MB25
02F8	MB26
0310	MB27
0328	MB28
0340	MB29
0358	MB30
0370	MB31
0388	MB32
03A0	MB33
03B8	MB34
03D0	MB35
03E8	MB36
0400	MB37
0418	MB38
0430	MB39
0448	MB40
0460	MB41
0480	MB42
0498	MB43
04B0	MB44
04C8	MB45
04E0	MB46
04F8	MB47
0510	MB48
0528	MB49
0540	MB50
0558	MB51
0570	MB52

Table continues on the next page...

Table 453. 16-byte message buffers (continued)

Address offset (hex)	MBDSR=b01 16-byte payload
0588	MB53
05A0	MB54
05B8	MB55
05D0	MB56
05E8	MB57
0600	MB58
0618	MB59
0630	MB60
0648	MB61
0660	MB62

Table 454. 32-byte message buffers

Address offset (hex)	MBDSR=b10 32-byte payload
0080	MB0
00A8	MB1
00D0	MB2
00F8	MB3
0120	MB4
0148	MB5
0170	MB6
0198	MB7
01C0	MB8
01E8	MB9
0210	MB10
0238	MB11
0280	MB12
02A8	MB13
02D0	MB14
02F8	MB15
0320	MB16

Table continues on the next page...

Table 454. 32-byte message buffers (continued)

Address offset (hex)	MBDSR=b10 32-byte payload
0348	MB17
0370	MB18
0398	MB19
03C0	MB20
03E8	MB21
0410	MB22
0438	MB23
0480	MB24
04A8	MB25
04D0	MB26
04F8	MB27
0520	MB28
0548	MB29
0570	MB30
0598	MB31
05C0	MB32
05E8	MB33
0610	MB34
0638	MB35

Table 455. 64-byte message buffers

Address offset (hex)	MBDSR=b11 64-byte payload
0080	MB0
00C8	MB1
0110	MB2
0158	MB3
01A0	MB4
01E8	MB5
0230	MB6
0280	MB7

Table continues on the next page...

Table 455. 64-byte message buffers (continued)

Address offset (hex)	MBDSR=b11 64-byte payload
02C8	MB8
0310	MB9
0358	MB10
03A0	MB11
03E8	MB12
0430	MB13
0480	MB14
04C8	MB15
0510	MB16
0558	MB17
05A0	MB18
05E8	MB19
0630	MB20

70.6.6 Legacy Rx FIFO structure

When MCR[RFEN] is set, the memory area 80h–DCh (which is normally occupied by MBs 0–5) is used by the reception Legacy FIFO engine.

The region 80h–8Ch contains the output of the Legacy Rx FIFO which must be read by the CPU as a message buffer. This output contains the oldest message that has been received but not yet read. The region 90h–DCh is reserved for internal use of the Legacy Rx FIFO engine.

An additional memory area, which starts at E0h and may extend up to 2DCh (normally occupied by MBs 6–37) depending on the CTRL2[RFFN] field setting, contains the ID filter table (configurable from 8 to 128 table elements) that specifies filtering criteria for accepting frames into the Legacy Rx FIFO.

Out of reset, the ID filter table flexible memory area defaults to E0h and extends only to FCh, which corresponds to MBs 6 to 7 for RFFN = 0, for backward compatibility with previous versions of FlexCAN.

The following shows the Legacy Rx FIFO data structure.

Table 456. Legacy Rx FIFO structure

	31 28 24	23 22	21	20	19 18 17 16	15 8	7 0
80h	IDHIT	SRR	IDE	RTR	DLC	TIME STAMP	
84h		ID standard			ID extended		
88h	Data byte 0	Data byte 1				Data byte 2	Data byte 3
8Ch	Data byte 4	Data byte 5				Data byte 6	Data byte 7

Table continues on the next page...

Table 456. Legacy Rx FIFO structure (continued)

90h–DCh	Reserved
E0h	ID filter table element 0
E4h	ID filter table element 1
E8h–2D4h	ID filter table elements 2 to 125
2D8h	ID filter table element 126
2DCh	ID filter table element 127
	= Unimplemented or reserved

Each ID filter table element occupies an entire 32-bit word and can be compounded by one, two, or four Identifier Acceptance Filters (IDAF) depending on the MCR[IDAM] field setting. The following figures show the IDAF indexation.

The following table shows the three different formats of the ID table elements. Note that all elements of the table must have the same format. See Legacy Rx FIFO for more information.

Table 457. ID table structure

Format	31	30	29 24	23 16	15	14	13 8	7 1	0
A	RTR	IDE	RXIDA (standard = 29–19, extended = 29–1)						
B	RTR	IDE	RXIDB_0 (standard = 29–19, extended = 29–16)		RTR	IDE	RXIDB_1 (standard = 13–3, extended = 13–0)		
C	RXIDC_0 (std/ext = 31–24)		RXIDC_1 (std/ext = 23–16)		RXIDC_2 (std/ext = 15–8)		RXIDC_3 (std/ext = 7–0)		
	= Unimplemented or Reserved								

RTR — Remote frame

This bit specifies if remote frames are accepted into the Legacy FIFO if they match the target ID.

1 = Remote frames can be accepted and data frames are rejected

0 = Remote frames are rejected and data frames can be accepted

IDE — Extended frame

Specifies whether extended or standard frames are accepted into the Legacy FIFO if they match the target ID.

1 = Extended frames can be accepted and standard frames are rejected

0 = Extended frames are rejected and standard frames can be accepted

RXIDA — Rx frame identifier (Format A)

Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In the standard frame format, only the 11 most significant bits (29 to 19) are used for frame identification. In the extended frame format, all bits are used.

RXIDB_0, RXIDB_1 — Rx frame identifier (Format B)

Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In the standard frame format, the 11 most significant bits (a full standard ID) (29 to 19 and 13 to 3) are used for frame identification. In the extended frame format, all 14 bits of the field are compared to the 14 most significant bits of the received ID.

RXIDC_0, RXIDC_1, RXIDC_2, RXIDC_3 — Rx frame identifier (Format C)

Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In both standard and extended frame formats, all 8 bits of the field are compared to the 8 most significant bits of the received ID.

IDHIT — Identifier acceptance filter hit indicator

This 9-bit field indicates which identifier acceptance filter was hit by the received message that is in the output of the Legacy Rx FIFO. See Legacy Rx FIFO for more information.

70.6.7 Enhanced Rx FIFO structure

When ERFCR[ERFEN] is set, the Enhanced Rx FIFO is enabled. The region 2000h–204Ch contains the output of the Enhanced Rx FIFO which must be read by the CPU as a message buffer. This output contains the oldest message that has been received but not yet read. The following table shows the Enhanced Rx FIFO data structure.

Table 458. Enhanced RX FIFO Structure

	31	30	29	28 … 24	23	22	21	20	19	18	17	16	15 … 8	7	6 … 0
2000h	EDL	BRS	ESI	RESERVED		SRR	IDE	RTR	DLC				TIME STAMP LEGACY		
2004h	RESERVED			ID (STANDARD/EXTENDED)									ID (EXTENDED)		
2008h	Data byte 0					Data byte 1							Data byte 2		Data byte 3
200Ch	Data byte 4					Data byte 5							Data byte 6		Data byte 7
2010h	Data byte 8					Data byte 9							Data byte 10		Data byte 11
2014h	Data byte 12					Data byte 13							Data byte 14		Data byte 15
2018h	Data byte 16					Data byte 17							Data byte 18		Data byte 19
201Ch	Data byte 20					Data byte 21							Data byte 22		Data byte 23
2020h	Data byte 24					Data byte 25							Data byte 26		Data byte 27
2024h	Data byte 28					Data byte 29							Data byte 30		Data byte 31
2028h	Data byte 32					Data byte 33							Data byte 34		Data byte 35
202Ch	Data byte 36					Data byte 37							Data byte 38		Data byte 39
2030h	Data byte 40					Data byte 41							Data byte 42		Data byte 43
2034h	Data byte 44					Data byte 45							Data byte 46		Data byte 47
2038h	Data byte 48					Data byte 49							Data byte 50		Data byte 51
203Ch	Data byte 52					Data byte 53							Data byte 54		Data byte 55
2040h	Data byte 56					Data byte 57							Data byte 58		Data byte 59
2044h	Data byte 60					Data byte 61							Data byte 62		Data byte 63
IH_OFF	RESERVED														ID HIT

Table continues on the next page...

Table 458. Enhanced RX FIFO Structure (continued)

TS_OFF	HR TIME STAMP
2050h	19 Enhanced FIFO Elements (RESERVED)
...	
263Ch	

NOTE

ID HIT offset and high resolution time stamp offset are changed dynamically according to data length code (DLC) as shown in Table 459.

Table 459. ID HIT offset and high resolution time stamp offset

Data Length Code (DLC)	ID HIT offset (IH_OFF)	High resolution time stamp offset (TS_OFF)
0	2008h	200Ch
1–4	200Ch	2010h
5–8	2010h	2014h
9	2014h	2018h
10	2018h	201Ch
11	201Ch	2020h
12	2020h	2024h
13	2028h	202Ch
14	2038h	203Ch
15	2048h	204Ch

EDL — Extended Data Length

This bit distinguishes between classical CAN format and CAN FD format frames.

0 – Classical CAN frame format

1 – CAN FD frame format

BRS — Bit Rate Switch

This bit defines whether the bit rate is switched inside a CAN FD format frame.

0 – Bit rate is not switched in a CAN FD frame

1 – Bit rate is switched in a CAN FD frame

ESI — Error State Indicator

This bit indicates if the transmitting node is error active or error passive. This bit is meaningful only if EDL = 1.

0 – Transmitting node is error active

1 – Transmitting node is error passive

SRR — Substitute Remote Request

Fixed recessive bit, used only in extended format. Transmitting nodes always send it as recessive and receiving nodes can receive it as either recessive or dominant. If FlexCAN receives this bit as dominant, then it is interpreted as an arbitration loss.

IDE — ID Extended Bit

This field identifies whether the frame format is standard or extended.

0 – Frame format is standard

1 – Frame format is extended

RTR — Remote Frame

0 = Indicates the current frame is a data frame

1 = Indicates the current frame is remote request

DLC — Data Length Code

This 4-bit field defines the number of bytes in the data field of a CAN frame (Data byte 0 to Data byte 63). When RTR = 1, the frame is a remote request and does not include the data field, regardless of the DLC field. Refer to Table 449 for more details.

LEGACY TIME STAMP — 16-bit Time Stamp

This 16-bit field is a copy of the Free-Running Timer, captured during the CAN frame. See Table 448 to find more details about legacy time stamp operation.

ID — Frame Identifier

In base frame format, only the 11 most significant bits are used for frame identification. The 18 least significant bits are ignored. In extended frame format, all bits are used for frame identification.

DATA BYTE 0 to 63 — Data Field

Up to sixty four bytes can be stored in the data field.

IDHIT — Identifier Acceptance Filter Hit Indicator

This field indicates which Enhanced Rx FIFO Filter Element (ERFFELn) was hit by the received message that is in the output of the Enhanced Rx FIFO. For each filter region, standard-ID filter space and extended-ID filter space, there is an independent index starting from zero. Table 460 shows how IDHIT is written by FlexCAN according to each filter element.

Table 460. IDHIT for Enhanced Rx FIFO

Enhanced Rx FIFO filter element — ERFFEL	ID HIT value	Filter element type
ERFFEL0	0	Extended-ID
ERFFEL1	1	Extended-ID
. . .	. . .	Extended-ID
ERFFEL(m–1)	m–1	Extended-ID
ERFFEL(m)	0	Standard-ID
ERFFEL(m+1)	1	Standard-ID
. . .	. . .	Standard-ID
ERFFEL(2n–m+1)	2x(n–m)+1	Standard-ID

NOTE

Where m = NEXIF and n = NFE. If NEXIF = 0, only standard-ID filter elements exist, and if NEXIF > NFE, then only extended-ID filter elements exist.

HR TIME STAMP — High-resolution Time Stamp

32-bit time base captured during the CAN frame. When CTRL2[TSTAMPCAP] is different from zero, a 32-bit time base is captured from a dedicated on-chip timer which operates in free running mode. See TSTAMPCAP field description in Control 2 Register (CTRL2) to find more details about capture point configuration of the high-resolution time stamp.

70.7 Glossary

Active message buffer	A message buffer is *active* if it can participate in the current matching or arbitration process.
Bus interface unit	FlexCAN submodule responsible for the interface to the CPU.
Bus off	See CAN Specification.[12]
CAN	Controller Area Network, a serial communication protocol defined in CAN Specification[12] and ISO International Standard.[13]
CHI	Controller-host interface, a FlexCAN submodule responsible for message buffer matching and arbitration algorithms.
CRC	Cyclic redundancy check
Dominant bit	A dominant bit wins the arbitration on the CAN bus. It is transmitted as 0.
Doze mode	A system low-power mode where the CPU bus remains active and a global Doze mode request is sent to all peripherals asking them to enter low power mode.
Error active	See CAN Specification.[12]
Error frame	See CAN Specification.[12]
Error passive	See CAN Specification.[12]
Form error	See CAN Specification.[12]
Hard reset	A reset coming from an external pin and/or following power-on. It resets everything.
Idle	See CAN Specification.[12]
Information processing time	See CAN Specification.[12]
Intermission	See CAN Specification.[12]
Matching elements	Data used in the matching process, such as ID, filter, mask, etc.
MB	See Message buffer.
Message buffer	Internal FlexCAN data structure containing bytes received from, or to be transmitted to, the CAN line, as well as information about this data.
Overload frame	See CAN Specification.[12]
Phase buffer segment	See CAN Specification.[12]
Recessive bit	A recessive bit loses the arbitration on the CAN bus. It is transmitted as 1.
Sclock	Serial clock. Obtained by dividing the clock feeding the CAN engine (oscillator or bus clock) by a prescaler factor. The Sclock period defines the time quantum for CAN protocol timing.

[12] Controller Area Network – CAN Specification Version 2.0 Part A, Part B, Robert Bosch GmbH, 1991.
[13] ISO International Standard – ISO 11898 First Edition 1993 Road Vehicles – Interchange of Digital Information – Controller Area Network (CAN) for high-speed Communication.

Soft reset	Global reset typically used by peripherals to reinitialize some of its registers, but not all of them.
Stop mode	A system low-power mode in which all chip clocks are stopped for maximum power savings.
Stuffing error	See CAN Specification.[12]
Time quantum	This is equal to the Sclock period. It is the minimum time period used to compose the CAN protocol bit timing.

Chapter 71
Synchronous Audio Interface (SAI)

71.1 Chip-specific SAI information

71.1.1 SAI instances and configuration

Table 461. SAI instances

Instance	MWCT2D16S/MWCT2D17S	MWCT2015S/ MWCT2016S
SAI_0	Yes	No
SAI_1	Yes	No

Table 462. SAI configuration

Chip	Instances	Synchronous operation between instances	DMA support	Data Rate/ channel	No. of Channels	Frame Synchronization	Master Clock Frequency	FIFO size(TX/RX)/ Channel	No. of words/ frame
MWCT2D16S, MWCT2D17S	SAI_0 SAI_1	No	Yes	12.288 Mbps	4 1	I2S, AC97, and CODEC/DSP interfaces	24.576 external audio	8X32 bit	8-16 words

NOTE

The SAI_BCLK (SAI_TX_BCLK, SAI_RX_BCLK), SAI_SYNC (SAI_TX_SYNC, SAI_RX_SYNC), and SAI_DATA (SAI_TX_DATA, SAI_RX_DATA) pins are shared between each module instance's (SAI_0, SAI_1) receiver and transmitter. See the IOMUX file attached to this document for details. At any specific time, either the SAI receiver or the SAI transmitter should be active or they should operate synchronously (only one active at a time). The SAI transmitter and SAI receiver should not be configured to operate simultaneously with inconsistent configurations.

NOTE

Timing specification of SAI master and SAI slave may limit maximum frequency.

71.2 Overview

The Synchronous Audio Interface (SAI) provides an interface that supports full-duplex serial interfaces with frame synchronization formats such as I^2S, AC97, TDM, and codec/DSP interfaces.

71.2.1 Block diagram

The following block diagram also shows the module clocks.

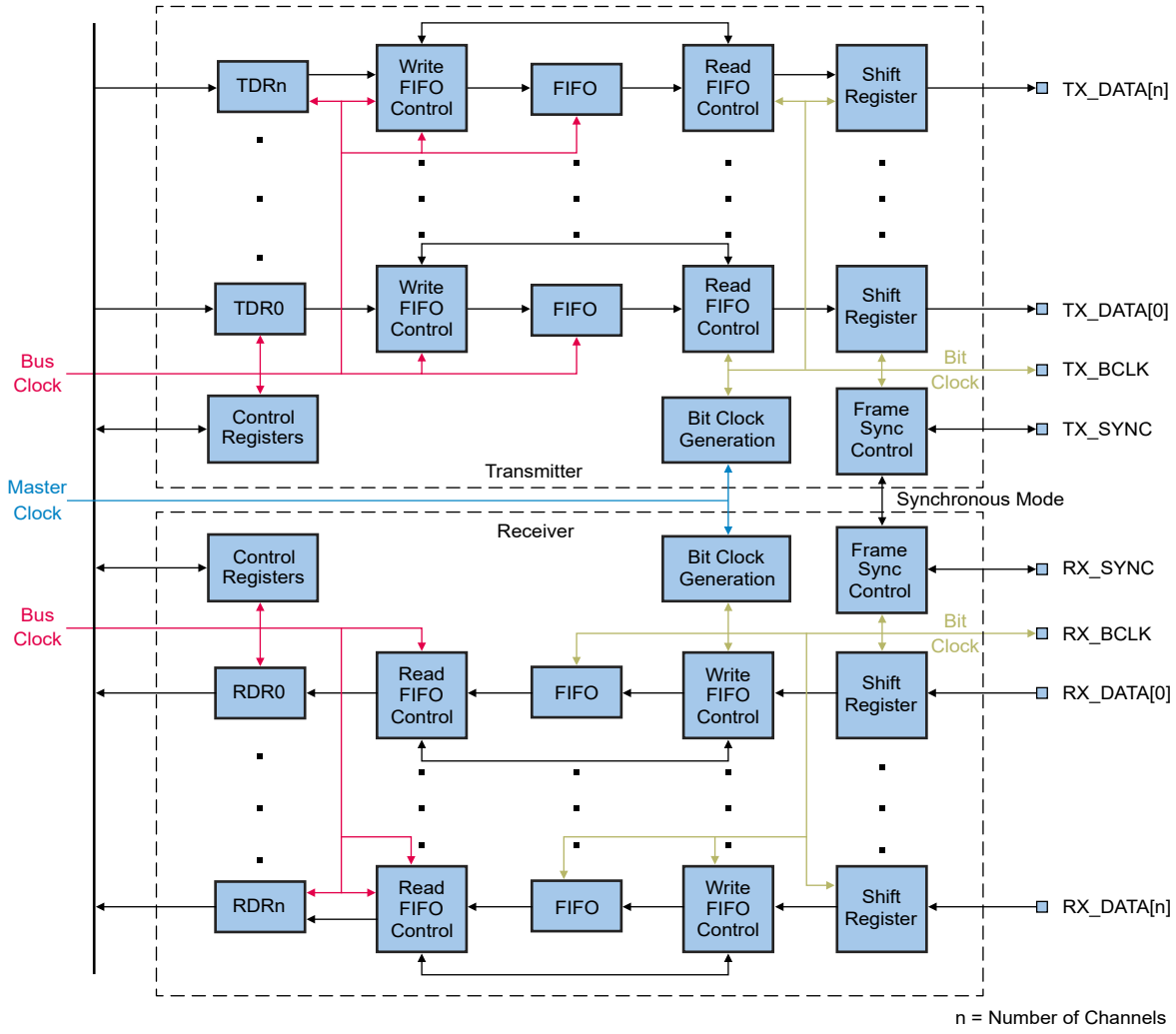


Figure 387. SAI block diagram

71.2.2 Features

Some of the features are not supported across all SAI instances; see the chip-specific SAI information in the first section of this chapter.

- Transmitter with independent bit clock and frame sync supporting 4 data lines
- Receiver with independent bit clock and frame sync supporting 4 data lines
- Each data line can support a maximum frame size of 16 words
- 8- to 32-bit word size
- Word size configured separately for the first word and remaining words in a frame
- Asynchronous 8 x 32-bit FIFO for each transmit and receive data line supporting:
  - Graceful restart after FIFO error
  - Automatic restart after FIFO error without software intervention
  - 8-bit and 16-bit data packing into each 32-bit FIFO word
  - Multiple data line FIFOs combining into single data line FIFO

71.3 Functional description

This section provides a complete functional description of the block.

71.3.1 Modes of operation

Module power modes include Run mode, and Debug mode.

Run mode

In Run mode, the SAI transmitter and receiver operate normally.

Debug mode

In Debug mode, the SAI transmitter and/or receiver continues operating if the Debug Enable bit is set. When TCSR[DBGE] or RCSR[DBGE] bit is clear and Debug mode is entered, the SAI is disabled after completing the current transmit or receive frame. Debug mode does not affect the transmitter and receiver bit clocks.

71.3.2 Synchronous modes

The SAI transmitter and receiver can operate synchronously to each other.

Synchronous mode

The SAI transmitter and receiver can be configured to operate with synchronous bit clock and frame sync.

If both the transmitter and receiver use the transmitter bit clock and frame sync:

- Configure the transmitter for asynchronous operation and the receiver for synchronous operation.
- Enable the receiver in synchronous mode only after both the transmitter and receiver are enabled.
- Enable the transmitter last and disable the tranmitter first.

If both the transmitter and receiver use the receiver bit clock and frame sync:

- Configure the receiver for asynchronous operation and the transmitter for synchronous operation.
- Enable the transmitter in synchronous mode only after both the receiver and transmitter are enabled.
- Enable the receiver last and disable the receiver first.

When operating in synchronous mode, the transmitter and receiver share only the bit clock, frame sync, and transmitter/receiver enable. Otherwise, the transmitter and receiver operate independently, although the configuration registers must be configured consistently across both the transmitter and receiver.

71.3.3 Frame sync configuration

When enabled, the SAI continuously transmits and/or receives frames of data. Each frame consists of a fixed number of words and each word consists of a fixed number of bits. Within each frame, any given word can be masked causing the receiver to ignore that word and the transmitter to tri-state for the duration of that word.

The frame sync signal indicates the start of each frame. A valid frame sync requires a rising edge (if active high) or falling edge (if active low) to be detected and the transmitter or receiver cannot be busy with a previous frame. A valid frame sync is also ignored (slave mode) or not generated (master mode) for the first four bit clock cycles after enabling the transmitter or receiver.

The transmitter and receiver frame sync can be configured independently with any of the following options:

- Generate externally or internally
- Require active high or active low
- Assert with the first bit in frame or assert one bit early
- Assert for a duration of between 1 bit clock and the first word length

- Set frame length from 1 to 16 words per frame
- Support word length of 8 to 32 bits per word
  - — First word length and remaining word lengths can be configured separately
- Transmit/Receive words MSB first or LSB first

These configuration options cannot be changed after enabling the SAI transmitter or receiver.

71.3.4 Data FIFO

Each transmit and receive channel includes a 8 x 32-bit size FIFO. To access the FIFO data use the SAI Transmit/Receive Data Registers.

71.3.4.1 Data alignment

Data in the FIFO can be aligned anywhere within the 32-bit wide register through the use of the First Bit Shifted configuration field, which selects the bit index (between 31 and 0) of the first bit shifted.

Examples of supported data alignment and the required First Bit Shifted configuration are illustrated in Figure 388 for LSB First configurations and Figure 389 for MSB First configurations.

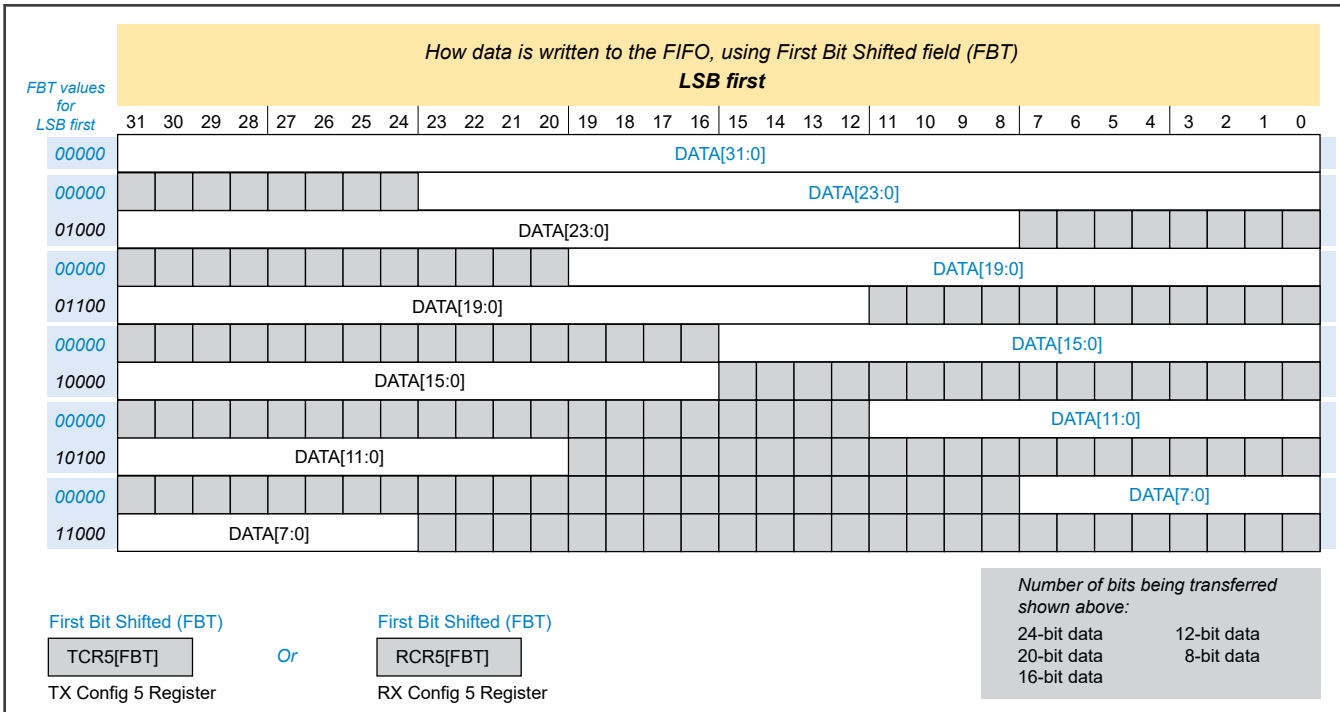


Figure 388. SAI first bit shifted, LSB first

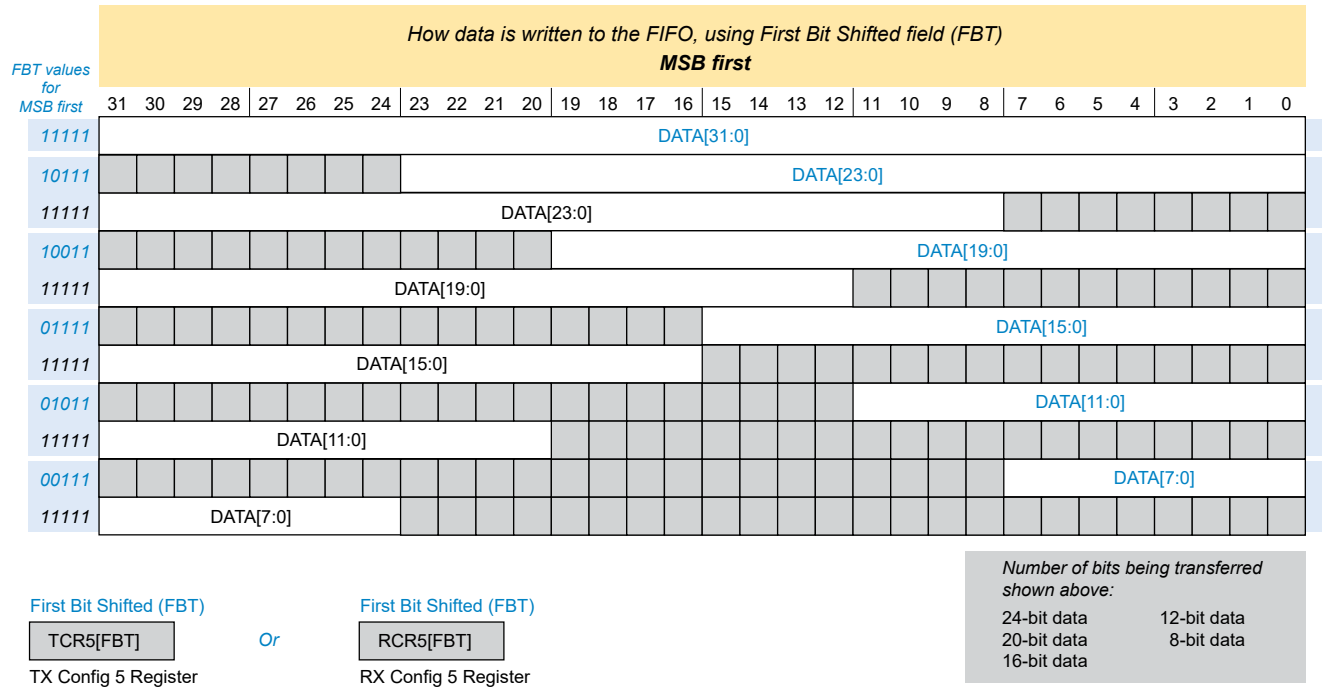


Figure 389. SAI first bit shifted, MSB first

71.3.4.2 FIFO pointers

When writing to a Transmit Data Register (TDR*n*), the Write FIFO Pointer (WFP) of the corresponding Transmit FIFO Register (TFR*n*) increments after each valid write. The SAI supports 8-bit, 16-bit, and 32-bit writes to the Transmit Data Register and the FIFO pointer increments after each individual write. Note that 8-bit writes should only be used when transmitting up to 8-bit data; 16-bit writes should only be used when transmitting up to 16-bit data.

- If the Transmit FIFO is full, writes to a Transmit Data Register are ignored.
- If the Transmit FIFO is empty, to avoid a FIFO underrun, write to the Transmit Data Register at least 3 bit clocks before the start of the next unmasked word. Before enabling the transmitter, initialize the Transmit FIFO with data (because after enabling the transmitter, the transmitter starts a new frame, and if no data is in the FIFO, then the transmitter immediately give an error).

When reading a Receive Data Register (RDR*n*), the Read FIFO Pointer (RFP) of the corresponding Receive FIFO Register (RFR*n*) increments after each valid read. The SAI supports 8-bit, 16-bit and 32-bit reads from the RDR and the FIFO pointer will increment after each individual read. Note that 8-bit reads should only be used when receiving up to 8-bit data; 16-bit reads should only be used when receiving up to 16-bit data.

- If the Receive FIFO is empty, reads from a Receive Data Register are ignored.
- If the Receive FIFO is full, to avoid a FIFO overrun, read the Receive Data Register at least 3 bit clocks before the end of an unmasked word.

71.3.4.3 FIFO packing

FIFO packing supports storing multiple 8-bit or 16-bit data words in one 32-bit FIFO word for the transmitter and/or receiver. While this can be emulated by adjusting the number of bits per word and number of words per frame (for example, one 32-bit word per frame versus two 16-bit words per frame), FIFO packing does not require even multiples of words per frame and fully supports word masking. When FIFO packing is enabled, the FIFO pointers only increment when the full 32-bit FIFO word has been written (transmit) or read (receive) by software, supporting scenarios where different words within each frame are loaded/stored in different areas of memory.

Using 16-bit FIFO packing for transmit, the transmit shift register loads at the start of each frame and after every second unmasked transmit word. The first word transmitted is taken from 16-bit word at byte offset 0x0 (the first bit is selected by TCR5[FBT] and must be configured within this 16-bit word) and the second word transmitted is taken from the 16-bit word at byte offset 0x2 (the first bit is selected by TCR5[FBT][3:0]). The transmitter transmits logic zero until the start of the next word once the 16-bit word has been transmitted.

Using 16-bit FIFO packing for receive, the receive shift register is stored after every second unmasked received word, and at the end of each frame if there is an odd number of unmasked received words in each frame. The first word received is stored in the 16-bit word at byte offset 0x0 (the first bit is selected by RCR5[FBT] and must be configured within this 16-bit word) and the second word received is stored in the 16-bit word at byte offset 0x2 (the first bit is selected by RCR5[FBT][3:0]). The receiver ignores received data until the start of the next word once the 16-bit word has been received.

The 8-bit FIFO packing is similar to 16-bit packing except four words are loaded or stored into each 32-bit FIFO word. The first word is loaded/stored in byte offset 0x0, second word in byte offset 0x1, third word in byte offset 0x2 and fourth word in byte offset 0x3. The TCR5[FBT] and/or RCR5[FBT] must be configured within byte offset 0x0.

71.3.4.4 FIFO Combine

FIFO Combining mode allows the separate FIFOs for multiple data channels to be used as a single FIFO for either software accesses or a single data channel or both. Note that the enabled data channels must be contiguous and data channel 0 must be enabled when using FIFO Combine mode.

Combining FIFOs for software access (writing transmit FIFO registers, reading receive FIFO registers) allows a DMA controller or software to read or write multiple FIFOs without incrementing the address that is accessed. Once enabled, the first software access to a FIFO register accesses the first enabled channel FIFO, while the second access to a FIFO register accesses the second enabled channel FIFO. This continues until software accesses the last enabled channel FIFO and the pointer resets back to the first enabled channel FIFO. To reset the pointer manually, software can reset the FIFOs or disable FIFO combining on software accesses.

Combining FIFOs for transmit data channels allows one data channel to use the FIFOs of all enabled channel FIFOs, with identical data output on each enabled data channel. The transmit shift registers for all enabled data channels load at the start of each frame and every N unmasked words (where N is the number of enabled data channels). The first word transmitted loads from the first enabled channel FIFO, while the second word transmitted loads from the second enabled channel FIFO, and so on until the end of the frame. Because the first word in each frame always loads from the first enabled data channel, it is recommended that the number of unmasked words in each frame is evenly divisible by the number of enabled data channels.

Combining FIFOs for receive data channels allows one data channel to use the FIFOs of all enabled channel FIFOs, with received data from channel 0 stored into each enabled data channel. The receive shift register for all enabled data channels are stored after every N unmasked words (where N is the number of enabled data channels). The first word received stores to the first enabled channel FIFO, while the second word received stores to the second enabled channel FIFO, and so on until the end of the frame. Because the first word in each frame always stores to the first enabled data channel, it is recommended that the number of unmasked words in each frame is evenly divisible by the number of enabled data channels.

Note that combining FIFOs for data channels loads or stores each channel FIFO at the same time. This means that FIFO error conditions are only checked every N words (where N is the number of enabled data channels) and that the FIFO warning and request flags assert if any of the enabled data channel meets the warning flag or request flag conditions.

71.3.5 Word mask register

The SAI transmitter and receiver each contain a word mask register (TMR and RMR) that can be used to mask any word in the frame. Because the word mask register is double buffered, software can update it before the end of each frame to mask a particular word in the next frame.

The TMR causes the Transmit Data pin to be tri-stated for the length of each selected word and the transmit FIFO is not read for masked words.

The RMR causes the received data for each selected word to be discarded and not written to the receive FIFO.

71.3.6 Interrupts and DMA requests

The SAI transmitter and receiver generate separate interrupts and separate DMA requests, but support the same status flags. Asynchronous interrupts are only generated when the system clock is gated provided the corresponding BCLK continues.

71.3.6.1 FIFO request flag

The FIFO request flag sets based on the number of entries in the FIFO and the FIFO watermark configuration.

The transmit FIFO request flag sets when the number of entries in any of the enabled transmit FIFOs is less than or equal to the transmit FIFO watermark configuration and clears when the number of entries in each enabled transmit FIFO is greater than the transmit FIFO watermark configuration.

The receive FIFO request flag sets when the number of entries in any of the enabled receive FIFOs is greater than the receive FIFO watermark configuration and clears when the number of entries in each enabled receive FIFO is less than or equal to the receive FIFO watermark configuration.

The FIFO request flag can generate an interrupt or a DMA request.

71.3.6.2 FIFO warning flag

The FIFO warning flag sets based on the number of entries in the FIFO.

The transmit warning flag sets when the number of entries in any of the enabled transmit FIFOs is empty and clears when the number of entries in each enabled transmit FIFO is not empty.

The receive warning flag sets when the number of entries in any of the enabled receive FIFOs is full and clears when the number of entries in each enabled receive FIFO is not full.

The FIFO warning flag can generate an Interrupt or a DMA request.

71.3.6.3 FIFO error flag

The Transmit FIFO Error Flag (TCSR[FEF] = 1]) sets when any of the enabled transmit FIFOs underrun. After the error flag sets, all enabled transmit channels transmit zero data before TCSR[FEF] clears.

When FIFO Continue on Error is enabled (TCR4[FCONT] = 1), the FIFO continues transmitting data following an underrun without software intervention. To ensure that data transmits in the correct order, the transmitter continues from the same word number in the frame that caused the FIFO to underrun, but only after new data writes to the transmit FIFO. Software should still clear the TCSR[FEF] flag, but without reinitializing the transmit FIFOs.

The Receive FIFO Error Flag sets (RCSR[FEF] = 1) when any of the enabled receive FIFOs overflow. After the flag is set, all enabled receive channels discard received data until RCSR[FEF] clears and the next receive frame starts. Empty all enabled receive FIFOs before RCSR[FEF] clears.

When Receive FIFO Continue on Error is enabled (RCR4[FCONT] = 1) the FIFO continues receiving data following an overflow without software intervention. To ensure that data is received in the correct order, the receiver continues from the same word number in the frame that caused the FIFO to overflow, but only after data has been read from the receive FIFO. Software should still clear the RCSR[FEF] flag, but without emptying the receive FIFOs.

The FIFO Error Flag can generate only an interrupt.

71.3.6.4 Sync error flag

The Sync Error Flag, TCSR[SEF] or RCSR[SEF], sets when configured for an externally generated frame sync and the external frame sync asserts when the transmitter or receiver is busy with the previous frame. The external frame sync assertion is ignored and the sync error flag sets. When the Sync Error Flag sets, the transmitter or receiver continues checking for frame sync assertion when idle or at the end of each frame.

The Sync Error Flag can only generate an interrupt.

71.3.6.5 Word start flag

The Word Start Flag sets (TCSR[WSF] = 1) at the start of the second bit clock for the selected word, as configured by the Word Flag Configuration field (TCR3[WDFL]).

The Word Start Flag can generate an interrupt only.

71.3.7 SAI clocking

The SAI clocks include:

- Audio master clock
- Bit clock
- Bus clock

Audio master clock

The audio master clock generates the bit clock when the receiver or transmitter is configured for an internally generated bit clock. The transmitter and receiver can independently select between the bus clock and up to three audio master clocks to generate the bit clock.

The audio master clock generation and selection is chip-specific. Refer to chip-specific clocking information about how the audio master clocks are generated.

Bit clock

The SAI transmitter and receiver support asynchronous free-running bit clocks that can be generated internally from an audio master clock or supplied externally. There is also the option for synchronous bit clock and frame sync operation between the receiver and transmitter or between multiple SAI peripherals.

- If both transmitter and receiver are configured for asynchronous operation, then the transmitter and receiver will each use *their own* bit clock and frame sync.
- If the *transmitter* is configured for asynchronous mode and the receiver is configured for synchronous mode, then both transmitter and receiver will use the *transmitter* bit clock and frame sync.
- If the *receiver* is configured for asynchronous mode and the transmitter is configured for synchronous mode, then both transmitter and receiver will use the *receiver* bit clock and frame sync.

Note that the software configures synchronous or asynchronous mode, and that choice selects the bit clock/frame sync used.

Externally generated bit clocks must be:

- Enabled before the SAI transmitter or receiver is enabled
- Disabled after the SAI transmitter or receiver is disabled and completes its current frames

If the SAI transmitter or receiver uses an externally generated bit clock in asynchronous mode and that bit clock is generated by an SAI that is disabled in stop mode, then the transmitter or receiver should be disabled by software before entering stop mode. This issue does not apply when the transmitter or receiver is in a synchronous mode because all synchronous SAIs are enabled and disabled simultaneously.

Bus clock

The bus clock is used by the control and configuration registers and to generate synchronous interrupts and DMA requests.

NOTE

Although there is no minimum bus clock frequency specified, the bus clock frequency must be fast enough (relative to the bit clock frequency) to ensure that the FIFOs can be serviced, without generating either a transmitter FIFO underrun or receiver FIFO overflow condition.

71.3.8 SAI resets

The SAI is asynchronously reset on system reset. The SAI has a software reset and a FIFO reset.

Software reset

The SAI transmitter includes a software reset that resets all transmitter internal logic, including the bit clock generation, status flags, and FIFO pointers. This software reset does not reset the configuration registers. The software reset remains asserted until cleared by software.

The SAI receiver includes a software reset that resets all receiver internal logic, including the bit clock generation, status flags and FIFO pointers. This software reset does not reset the configuration registers. The software reset remains asserted until cleared by software.

FIFO reset

The SAI transmitter includes a FIFO reset that synchronizes the FIFO write pointer to the same value as the FIFO read pointer. This FIFO reset empties the FIFO contents and is used after TCSR[FEF] is set, and before the FIFO is re-initialized and TCSR[FEF] is cleared. The FIFO reset is asserted for one cycle only.

The SAI transmitter can also reset the FIFO of individual data channels by setting the appropriate TCR3[CFR] bit. This should only be done when the corresponding TCR3[TCE] bit is clear.

The SAI receiver includes a FIFO reset that synchronizes the FIFO read pointer to the same value as the FIFO write pointer. This empties the FIFO contents and is to be used after the RCSR[FEF] is set and any remaining data has been read from the FIFO, and before the RCSR[FEF] is cleared. The FIFO reset is asserted for one cycle only.

The SAI receiver can also reset the FIFO of individual data channels by setting the appropriate RCR3[CFR] bit. This should only be done when the corresponding RCR3[RCE] bit is clear.

71.4 External signals

Name	Function	I/O
TX_BCLK	Transmit Bit Clock. The bit clock is an input when externally generated and an output when internally generated.	I/O
TX_SYNC	Transmit Frame Sync. The frame sync is an input sampled synchronously by the bit clock when externally generated and an output generated synchronously by the bit clock when internally generated.	I/O
TX_DATA[3:0]	Transmit Data. The transmit data is generated synchronously by the bit clock and is tristated whenever not transmitting a word.	O
RX_BCLK	Receive Bit Clock. The bit clock is an input when externally generated and an output when internally generated.	I/O
RX_SYNC	Receive Frame Sync. The frame sync is an input sampled synchronously by the bit clock when externally generated and an output generated synchronously by the bit clock when internally generated.	I/O

Table continues on the next page...

Table continued from the previous page...

Name	Function	I/O
RX_DATA[3:0]	Receive Data. The receive data is sampled synchronously by the bit clock.	I

71.5 Memory map and register definition

A read or write access to an address from offset 0x100 and above will result in a bus error.

71.5.1 SAI register descriptions

71.5.1.1 SAI memory map

SAI_0 base address: 4036_C000h

SAI_1 base address: 404D_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0301_0000h
4h	Parameter (PARAM)	32	R	See section
8h	Transmit Control (TCSR)	32	RW	0000_0000h
Ch	Transmit Configuration 1 (TCR1)	32	RW	0000_0000h
10h	Transmit Configuration 2 (TCR2)	32	RW	0000_0000h
14h	Transmit Configuration 3 (TCR3)	32	RW	0000_0000h
18h	Transmit Configuration 4 (TCR4)	32	RW	0000_0000h
1Ch	Transmit Configuration 5 (TCR5)	32	RW	0000_0000h
20h - 2Ch	Transmit Data (TDR0 - TDR3)	32	W	See section
40h - 4Ch	Transmit FIFO (TFR0 - TFR3)	32	R	See section
60h	Transmit Mask (TMR)	32	RW	0000_0000h
88h	Receive Control (RCSR)	32	RW	0000_0000h
8Ch	Receive Configuration 1 (RCR1)	32	RW	0000_0000h
90h	Receive Configuration 2 (RCR2)	32	RW	0000_0000h
94h	Receive Configuration 3 (RCR3)	32	RW	0000_0000h
98h	Receive Configuration 4 (RCR4)	32	RW	0000_0000h
9Ch	Receive Configuration 5 (RCR5)	32	RW	0000_0000h
A0h - ACh	Receive Data (RDR0 - RDR3)	32	R	See section
C0h - CCh	Receive FIFO (RFR0 - RFR3)	32	R	See section
E0h	Receive Mask (RMR)	32	RW	0000_0000h

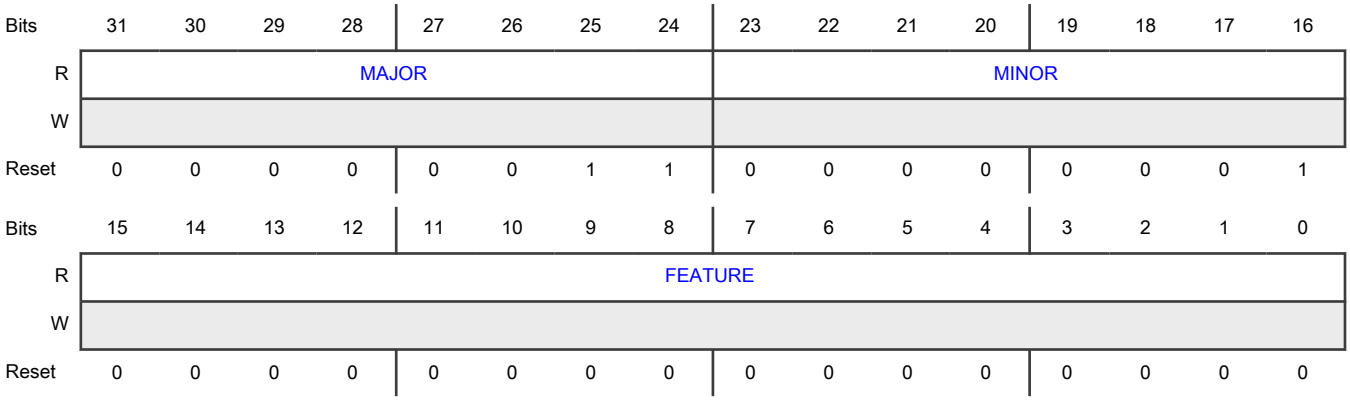
71.5.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function
Contains version numbers for the module design and feature set.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the specification.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000_0000_0000_0000b - Standard feature set.

71.5.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Contains parameter values that were implemented in the module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												FRAME			
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FIFO				0				DATALINE			
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PARAM	SAI_0: 0004_0304h SAI_1: 0004_0301h

Fields

Field	Function
31-20 —	Reserved
19-16 FRAME	Frame Size The maximum number of slots per frame is 2^FRAME.
15-12 —	Reserved
11-8 FIFO	FIFO Size The number of words in each FIFO is 2^FIFO.
7-4 —	Reserved
3-0 DATALINE	Number of Datalines The number of datalines implemented.

71.5.1.4 Transmit Control (TCSR)

Offset

Register	Offset
TCSR	8h

Function

Contains transmitter enable fields including resets, error and interrupt enable fields, and error flag fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TE	Reserved	DBGE	BCE	0		0	SR	0			WSF	SEF	FEF	FWF	FRF
W							FR					W1C	W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			WSIE	SEIE	FEIE	FWIE	FRIE	0			0			FWDE	FRDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TE	Transmitter Enable Enables/disables the transmitter. When software clears this field, the transmitter remains enabled, and this bit remains set, until the end of the current frame. 0b - Transmitter is disabled. 1b - Transmitter is enabled, or transmitter has been disabled and has not yet reached end of frame.
30 —	Reserved Software should only write zero to this reserved bit.
29 DBGE	Debug Enable Enables/disables transmitter operation in Debug mode. The transmit bit clock is not affected by debug mode. 0b - Transmitter is disabled in Debug mode, after completing the current frame. 1b - Transmitter is enabled in Debug mode.
28	Bit Clock Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
BCE	Enables the transmit bit clock, separately from the TE. This field is automatically set whenever TE is set. When software clears this field, the transmit bit clock remains enabled, and this bit remains set, until the end of the current frame. 0b - Transmit bit clock is disabled. 1b - Transmit bit clock is enabled.
27-26 —	Reserved
25 FR	FIFO Reset Empties the FIFO, and sets the FIFO read and write pointers to the same value, which may or may not be zero. Reading this field will always return zero. FIFO pointers should only be reset when the transmitter is disabled or the FIFO error flag is set. 0b - No effect. 1b - FIFO reset.
24 SR	Software Reset When set, resets the internal transmitter logic including the FIFO read and write pointers. Software-visible registers are not affected, except for the status registers. 0b - No effect. 1b - Software reset.
23-21 —	Reserved
20 WSF	Word Start Flag Indicates that the start of the configured word has been detected. Write a logic 1 to this field to clear this flag. 0b - Start of word not detected. 1b - Start of word detected.
19 SEF	Sync Error Flag Indicates that an error in the externally-generated frame sync has been detected. Write a logic 1 to this field to clear this flag. 0b - Sync error not detected. 1b - Frame sync error detected.
18 FEF	FIFO Error Flag Indicates that an enabled transmit FIFO has underrun. Write a logic 1 to this field to clear this flag. 0b - Transmit underrun not detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Transmit underrun detected.
17 FWF	FIFO Warning Flag Indicates that an enabled transmit FIFO is empty. 0b - No enabled transmit FIFO is empty. 1b - Enabled transmit FIFO is empty.
16 FRF	FIFO Request Flag Indicates that the number of words in an enabled transmit channel FIFO is less than or equal to the transmit FIFO watermark. 0b - Transmit FIFO watermark has not been reached. 1b - Transmit FIFO watermark has been reached.
15-13 —	Reserved
12 WSIE	Word Start Interrupt Enable Enables/disables word start interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
11 SEIE	Sync Error Interrupt Enable Enables/disables sync error interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
10 FEIE	FIFO Error Interrupt Enable Enables/disables FIFO error interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
9 FWIE	FIFO Warning Interrupt Enable Enables/disables FIFO warning interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
8 FRIE	FIFO Request Interrupt Enable Enables/disables FIFO request interrupts. 0b - Disables the interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enables the interrupt.
7-5 —	Reserved
4-2 —	Reserved
1 FWDE	FIFO Warning DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.
0 FRDE	FIFO Request DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.

71.5.1.5 Transmit Configuration 1 (TCR1)

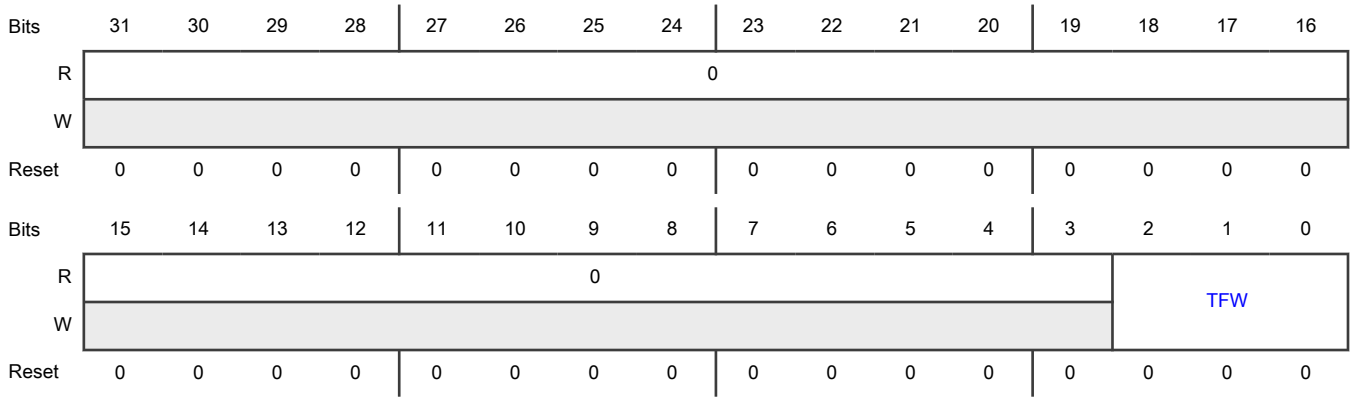
Offset

Register	Offset
TCR1	Ch

Function

Configures the watermark level for all enabled transmit channels.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 TFW	Transmit FIFO Watermark

71.5.1.6 Transmit Configuration 2 (TCR2)

Offset

Register	Offset
TCR2	10h

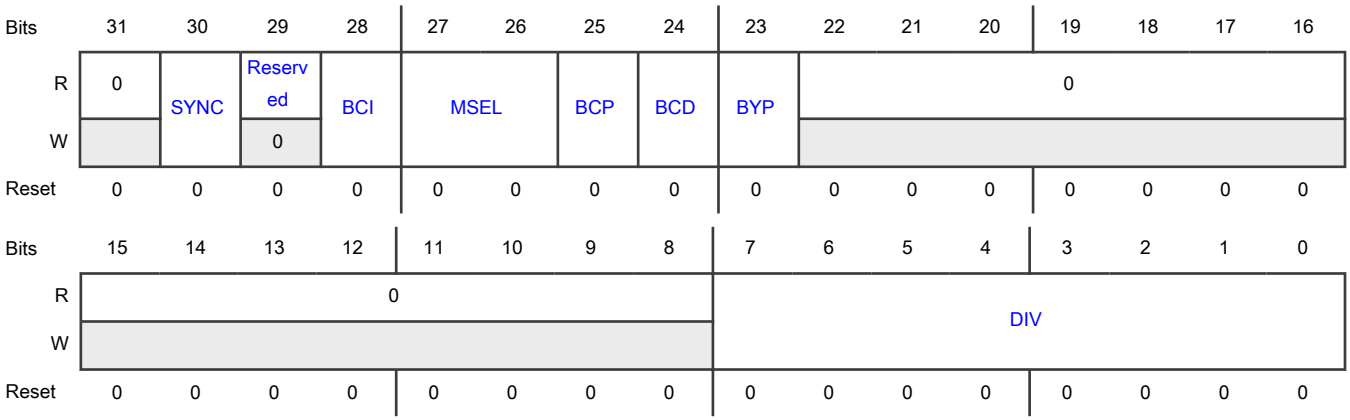
Function

Contains the SYNC mode and clock setting fields.

NOTE

This register must not be altered when TCSR[TE] is set.

Diagram



Fields

Field	Function
31 —	Reserved
30	Synchronous Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYNC	Configures between asynchronous and synchronous modes of operation. When configured for a synchronous mode of operation, the receiver must be configured for asynchronous operation. 0b - Asynchronous mode. 1b - Synchronous with receiver.
29 —	Reserved
28 BCI	Bit Clock Input When this field is set and using an internally generated bit clock in either synchronous or asynchronous mode, the bit clock actually used by the transmitter is delayed by the pad output delay (the transmitter is clocked by the pad input as if the clock was externally generated). This has the effect of decreasing the data input setup time, but increasing the data output valid time. The slave mode timing from the datasheet should be used for the transmitter when this bit is set. In synchronous mode, this bit allows the transmitter to use the slave mode timing from the datasheet, while the receiver uses the master mode timing. This field has no effect when configured for an externally generated bit clock . 0b - No effect. 1b - Internal logic is clocked as if bit clock was externally generated.
27-26 MSEL	MCLK Select Selects the audio Master Clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE** Depending on the device, some Master Clock options might not be available. See the chip-specific information for the meaning of each option. 00b - Bus Clock selected. 01b - Master Clock (MCLK) 1 option selected. 10b - Master Clock (MCLK) 2 option selected. 11b - Master Clock (MCLK) 3 option selected.
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. 0b - Bit clock is active high with drive outputs on rising edge and sample inputs on falling edge. 1b - Bit clock is active low with drive outputs on falling edge and sample inputs on rising edge.
24 BCD	Bit Clock Direction Configures the direction of the bit clock. 0b - Bit clock is generated externally in Slave mode. 1b - Bit clock is generated internally in Master mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 BYP	Bit Clock Bypass Bypasses the bit clock divider, internal bit clock is divide by 1 of the audio master clock. 0b - Internal bit clock is generated from bit clock divider. 1b - Internal bit clock is divide by one of the audio master clock.
22-8 —	Reserved
7-0 DIV	Bit Clock Divide Divides down the audio master clock to generate the bit clock when configured for an internal bit clock. The division value is (DIV + 1) * 2.

71.5.1.7 Transmit Configuration 3 (TCR3)

Offset

Register	Offset
TCR3	14h

Function

Contains the transmit channel settings.

Diagram

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

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

Fields

Field	Function
31-28	Reserved

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>—</td><td></td></tr>
<tr><td>27-24
CFR</td><td>Channel FIFO Reset
Resets the FIFO pointers for a specific channel. Reading this field always returns zero. FIFO pointers should only be reset when a channel is disabled or the FIFO error flag is set.
The width of CFR field = the number of transmit channels (call it N). For example, if CFR is 2 bits wide, then bit position 24 refers to transmit channel 1 FIFO pointer and bit position 25 refers to transmit channel 2 FIFO pointer. Setting bit 24 resets transmit channel 1 FIFO pointer, and setting bit 25 enables transmit channel 2 FIFO pointer. Setting bit N will reset transmit channel N FIFO pointer.
NOTE
When there is only a single channel, there is no need for individual channel FIFO reset (TCR3[CFR]). In the case of a single channel, use the global FIFO reset (TCSR[FR]).
0b - No effect.
1b - Transmit data channel N FIFO is reset.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI_0</td><td>TCR3</td><td>—</td></tr><tr><td>SAI_1</td><td>—</td><td>TCR3</td></tr></table></td></tr>
<tr><td>23-20
—</td><td>Reserved</td></tr>
<tr><td>19-16
TCE</td><td>Transmit Channel Enable
Enables the corresponding data channel for transmit operation. Changing TCE field will take effect immediately for generating the FIFO request and warning flags, but at the end of each frame for transmit operation.
The width of TCE field = the number of transmit channels (call it N). For example, if TCE field is 2 bits wide, then bit position 16 refers to transmit channel 1 and bit position 17 refers to transmit channel 2. Setting bit 16 enables transmit channel 1, and setting bit 17 enables transmit channel 2. Setting bit N will enable transmit channel N.
0b - Transmit data channel N is disabled.
1b - Transmit data channel N is enabled.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI_0</td><td>TCR3</td><td>—</td></tr><tr><td>SAI_1</td><td>TCR3[16]</td><td>TCR3[19–17]</td></tr></table>
15-4 —	Reserved
3-0 WDFL	Word Flag Configuration Configures which word sets the start of word flag. The value written must be one less than the word number. For example, writing 0 configures the first word in the frame. When configured to a value greater than TCR4[FRSZ], then the start of word flag is never set.

71.5.1.8 Transmit Configuration 4 (TCR4)

Offset

Register	Offset
TCR4	18h

Function

Contains the transmit fields for FIFO Combine Mode, FIFO Packing Mode, and frame sync settings.

NOTE

This register must not be altered when TCSR[TE] is set.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			SYWD					0		CHMOD	MF	FSE	ONDEM	FSP	FSD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31-29
—</td><td>Reserved</td></tr>
<tr><td>28
FCONT</td><td>FIFO Continue on Error
Configures when the SAI will continue transmitting after a FIFO error has been detected.
0b - On FIFO error, the SAI will continue from the start of the next frame after the FIFO error flag has been cleared.
1b - On FIFO error, the SAI will continue from the same word that caused the FIFO error to set after the FIFO warning flag has been cleared.</td></tr>
<tr><td>27-26
FCOMB</td><td>FIFO Combine Mode
When FIFO combine mode is enabled for FIFO writes, software writing to any FIFO data register will alternate the write among the enabled data channel FIFOs. For example, if two data channels are enabled then the first write will be performed to the first enabled data channel FIFO and the second write will be performed to the second enabled data channel FIFO. Reseting the FIFO or disabling FIFO combine mode for FIFO writes will reset the pointer back to the first enabled data channel.
When FIFO combine mode is enabled for FIFO reads from the transmit shift registers, the transmit data channel output will alternate between the enabled data channel FIFOs. For example, if two data channels are enabled then the first unmasked word will be transmitted from the first enabled data channel FIFO and the second unmasked word will be transmitted from the second enabled data channel FIFO. Since the first word of the frame is always transmitted from the first enabled data channel FIFO, it is recommended that the number of unmasked words per frame is evenly divisible by the number of enabled data channels.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI_0</td><td>TCR4</td><td>—</td></tr><tr><td>SAI_1</td><td>—</td><td>TCR4</td></tr></table>
00b - FIFO combine mode disabled.
01b - FIFO combine mode enabled on FIFO reads (from transmit shift registers).
10b - FIFO combine mode enabled on FIFO writes (by software).
11b - FIFO combine mode enabled on FIFO reads (from transmit shift registers) and writes (by software).</td></tr>
<tr><td>25-24
FPACK</td><td>FIFO Packing Mode
Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8-bit or 16-bit then only the first 8-bit or 16-bits are loaded from the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO write pointer will only increment when the full 32-bit FIFO word has been written by software.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - FIFO packing is disabled. 01b - Reserved 10b - 8-bit FIFO packing is enabled. 11b - 16-bit FIFO packing is enabled.
23-20 —	Reserved
19-16 FRSZ	Frame size Configures the number of words in each frame. The value written must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 16 words.
15-13 —	Reserved
12-8 SYWD	Sync Width Configures the length of the frame sync in number of bit clocks. The value written must be one less than the number of bit clocks. For example, write 0 for the frame sync to assert for one bit clock only. The sync width cannot be configured longer than the first word of the frame.
7-6 —	Reserved
5 CHMOD	Channel Mode Configures if transmit data pins are configured for TDM mode or Output mode. 0b - TDM mode, transmit data pins are tri-stated when slots are masked or channels are disabled. 1b - Output mode, transmit data pins are never tri-stated and will output zero when slots are masked or channels are disabled.
4 MF	MSB First Configures whether the LSB or the MSB is transmitted first. 0b - LSB is transmitted first. 1b - MSB is transmitted first.
3 FSE	Frame Sync Early 0b - Frame sync asserts with the first bit of the frame. 1b - Frame sync asserts one bit before the first bit of the frame.
2 ONDEM	On Demand Mode When set, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is clear.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Internal frame sync is generated continuously. 1b - Internal frame sync is generated when the FIFO warning flag is clear.
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync. 0b - Frame sync is active high. 1b - Frame sync is active low.
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0b - Frame sync is generated externally in Slave mode. 1b - Frame sync is generated internally in Master mode.

71.5.1.9 Transmit Configuration 5 (TCR5)

Offset

Register	Offset
TCR5	1Ch

Function

Contains transmit word width and bit index settings.

NOTE

This register must not be altered when TCSR[TE] is set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				WNW				0				W0W			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31-29 —	Reserved
28-24 WNW	Word N Width Configures the number of bits in each word, for each word except the first in the frame. The value written must be one less than the number of bits per word. Word width of less than 8 bits is not supported.
23-21 —	Reserved
20-16 W0W	Word 0 Width Configures the number of bits in the first word in each frame. The value written must be one less than the number of bits in the first word. Word width of less than 8 bits is not supported if there is only one word per frame.
15-13 —	Reserved
12-8 FBT	First Bit Shifted Configures the bit index for the first bit transmitted for each word in the frame. If configured for MSB First, the index of the next bit transmitted is one less than the current bit transmitted. If configured for LSB First, the index of the next bit transmitted is one more than the current bit transmitted. The value written must be greater than or equal to the word width when configured for MSB First. The value written must be less than or equal to 31-word width when configured for LSB First.
7-0 —	Reserved

71.5.1.10 Transmit Data (TDR0 - TDR3)

Offset

Register	Offset
TDR0	20h
TDR1	24h
TDR2	28h
TDR3	2Ch

Function

Writes to this register when the transmit FIFO is not full will push the data written into the transmit data FIFO. Writes to this register when the transmit FIFO is full are ignored.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
SAI_0	TDR0–TDR3	—
SAI_1	TDR0	TDR1–TDR3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	TDR															
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	TDR															
Reset	See Register reset values.															

Register reset values

Register	Reset value
TDR0	SAI_0,SAI_1: 0000_0000h
TDR1–TDR3	0000_0000h

Fields

Field	Function
31-0 TDR	Transmit Data Register

71.5.1.11 Transmit FIFO (TFR0 - TFR3)

Offset

Register	Offset
TFR0	40h
TFR1	44h
TFR2	48h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TFR3	4Ch

Function

The MSB of the read and write pointers is used to distinguish between FIFO full and empty conditions. If the read and write pointers are identical, then the FIFO is empty. If the read and write pointers are identical except for the MSB, then the FIFO is full.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
SAI_0	TFR0–TFR3	—
SAI_1	TFR0	TFR1–TFR3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WCP	0											WFP			
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												RFP			
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
TFR0	SAI_0,SAI_1: 0000_0000h
TFR1–TFR3	0000_0000h

Fields

Field	Function
31 WCP	Write Channel Pointer When FIFO Combine mode is enabled for writes, indicates that this data channel is the next FIFO to be written.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SAI_0</td><td>TFR0–TFR3</td><td>—</td></tr>
<tr><td>SAI_1</td><td>—</td><td>TFR0</td></tr>
</table>
0b - No effect.
1b - FIFO combine is enabled for FIFO writes and this FIFO will be written on the next FIFO write.
</td></tr>
<tr><td>30-20
—</td><td>Reserved</td></tr>
<tr><td>19-16
WFP</td><td>Write FIFO Pointer
FIFO write pointer for transmit data channel.</td></tr>
<tr><td>15-4
—</td><td>Reserved</td></tr>
<tr><td>3-0
RFP</td><td>Read FIFO Pointer
FIFO read pointer for transmit data channel.</td></tr>
</table>

71.5.1.12 Transmit Mask (TMR)

Offset

Register	Offset
TMR	60h

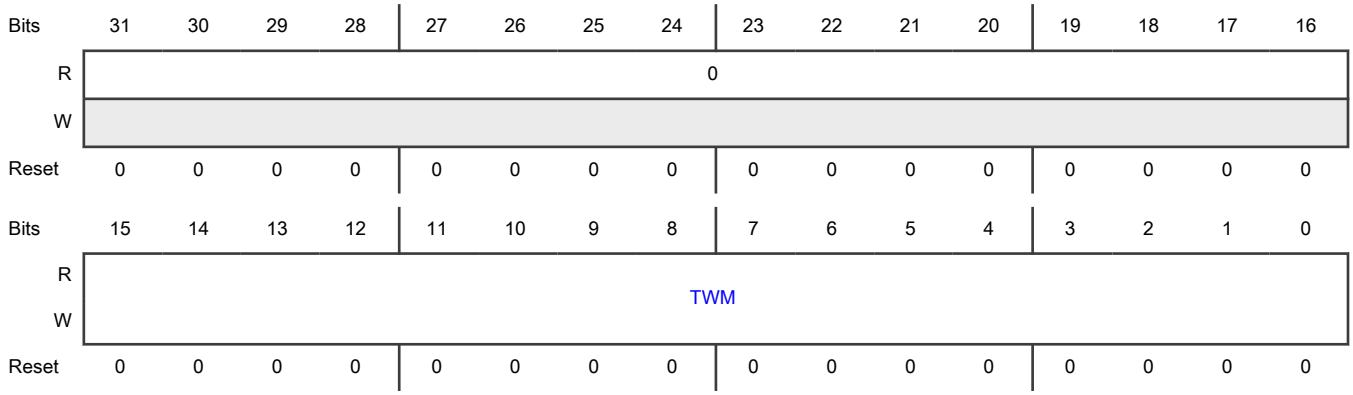
Function

This register is double-buffered and updates:

1. When TCSR[TE] is first set
2. At the end of each frame

This allows the masked words in each frame to change from frame to frame.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TWM	Transmit Word Mask Configures whether the transmit word is masked (transmit data pins are tri-stated or drive zero and transmit data not read from FIFO) for the corresponding word in the frame. 0000_0000_0000_0000b - Word N is enabled. 0000_0000_0000_0001b - Word N is masked. The transmit data pins are tri-stated or drive zero when masked.

71.5.1.13 Receive Control (RCSR)

Offset

Register	Offset
RCSR	88h

Function

Contains receiver enable fields including resets, error and interrupt enable fields, and error flag fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RE	Reserved	DBGE	BCE	0		0	SR	0			WSF	SEF	FEF	FWF	FRF
W							FR					W1C	W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			WSIE	SEIE	FEIE	FWIE	FRIE	0			0			FWDE	FRDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 RE	Receiver Enable Enables/disables the receiver. When software clears this field, the receiver remains enabled, and this bit remains set, until the end of the current frame. 0b - Receiver is disabled. 1b - Receiver is enabled, or receiver has been disabled and has not yet reached end of frame.
30 —	Reserved Software should only write zero to this reserved bit.
29 DBGE	Debug Enable Enables/disables receiver operation in Debug mode. The receive bit clock is not affected by Debug mode. 0b - Receiver is disabled in Debug mode, after completing the current frame. 1b - Receiver is enabled in Debug mode.
28 BCE	Bit Clock Enable Enables the receive bit clock, separately from RE. This field is automatically set whenever RE is set. When software clears this field, the receive bit clock remains enabled, and this field remains set, until the end of the current frame. 0b - Receive bit clock is disabled. 1b - Receive bit clock is enabled.
27-26 —	Reserved
25 FR	FIFO Reset Empties the FIFO, and sets the FIFO read and write pointers to the same value, which may or may not be zero. Reading this field will always return zero. FIFO pointers should only be reset when the receiver is disabled or the FIFO error flag is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect. 1b - FIFO reset.
24 SR	Software Reset Resets the internal receiver logic including the FIFO pointers. Software-visible registers are not affected, except for the status registers. 0b - No effect. 1b - Software reset.
23-21 —	Reserved
20 WSF	Word Start Flag Indicates that the start of the configured word has been detected. Write a logic 1 to this field to clear this flag. 0b - Start of word not detected. 1b - Start of word detected.
19 SEF	Sync Error Flag Indicates that an error in the externally-generated frame sync has been detected. Write a logic 1 to this field to clear this flag. 0b - Sync error not detected. 1b - Frame sync error detected.
18 FEF	FIFO Error Flag Indicates that an enabled receive FIFO has overflowed. Write a logic 1 to this field to clear this flag. 0b - Receive overflow not detected. 1b - Receive overflow detected.
17 FWF	FIFO Warning Flag Indicates that an enabled receive FIFO is full. 0b - No enabled receive FIFO is full. 1b - Enabled receive FIFO is full.
16 FRF	FIFO Request Flag Indicates that the number of words in an enabled receive channel FIFO is greater than the receive FIFO watermark. 0b - Receive FIFO watermark not reached. 1b - Receive FIFO watermark has been reached.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-13 —	Reserved
12 WSIE	Word Start Interrupt Enable Enables/disables word start interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
11 SEIE	Sync Error Interrupt Enable Enables/disables sync error interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
10 FEIE	FIFO Error Interrupt Enable Enables/disables FIFO error interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
9 FWIE	FIFO Warning Interrupt Enable Enables/disables FIFO warning interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
8 FRIE	FIFO Request Interrupt Enable Enables/disables FIFO request interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
7-5 —	Reserved
4-2 —	Reserved
1 FWDE	FIFO Warning DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.
0	FIFO Request DMA Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
FRDE	Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.

71.5.1.14 Receive Configuration 1 (RCR1)

Offset

Register	Offset
RCR1	8Ch

Function

Configures the watermark level for all enabled receiver channels.

Diagram

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

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

Fields

Field	Function
31-3 —	Reserved
2-0 RFW	Receive FIFO Watermark

71.5.1.15 Receive Configuration 2 (RCR2)

Offset

Register	Offset
RCR2	90h

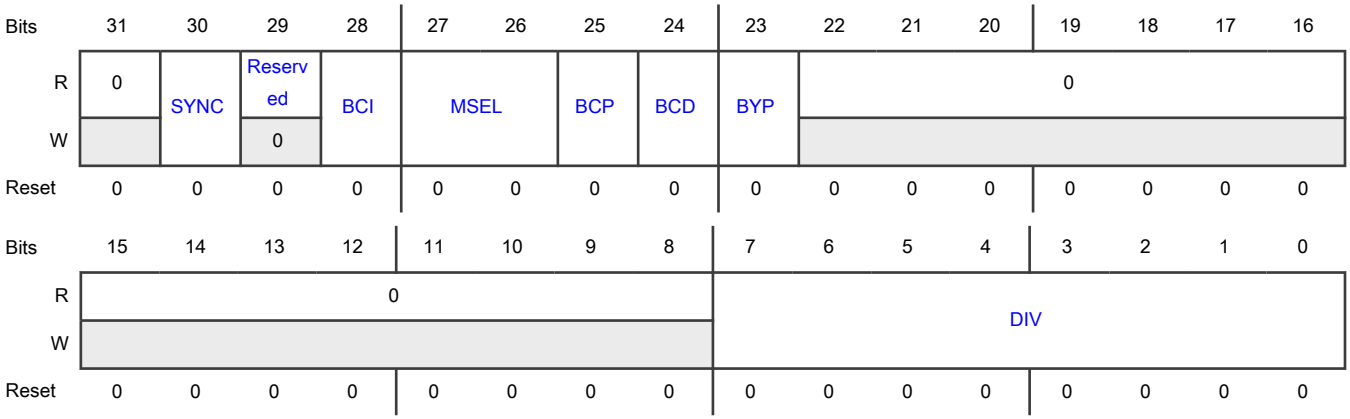
Function

Contains the SYNC mode and clock setting fields.

NOTE

This register must not be altered when RCSR[RE] is set.

Diagram



Fields

Field	Function
31 —	Reserved
30 SYNC	Synchronous Mode Configures between asynchronous and synchronous modes of operation. When configured for a synchronous mode of operation, the transmitter must be configured for asynchronous operation. 0b - Asynchronous mode. 1b - Synchronous with transmitter.
29 —	Reserved
28 BCI	Bit Clock Input

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is set and using an internally generated bit clock in either synchronous or asynchronous mode, the bit clock actually used by the receiver is delayed by the pad output delay (the receiver is clocked by the pad input as if the clock was externally generated). This has the effect of decreasing the data input setup time, but increasing the data output valid time. The slave mode timing from the datasheet should be used for the receiver when this bit is set. In synchronous mode, this bit allows the receiver to use the slave mode timing from the datasheet, while the transmitter uses the master mode timing. This field has no effect when configured for an externally generated bit clock . 0b - No effect. 1b - Internal logic is clocked as if bit clock was externally generated.
27-26 MSEL	MCLK Select Selects the audio Master Clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE** Depending on the device, some Master Clock options might not be available. See the chip-specific information for the availability and chip-specific meaning of each option. 00b - Bus Clock selected. 01b - Master Clock (MCLK) 1 option selected. 10b - Master Clock (MCLK) 2 option selected. 11b - Master Clock (MCLK) 3 option selected.
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. 0b - Bit Clock is active high with drive outputs on rising edge and sample inputs on falling edge. 1b - Bit Clock is active low with drive outputs on falling edge and sample inputs on rising edge.
24 BCD	Bit Clock Direction Configures the direction of the bit clock. 0b - Bit clock is generated externally in Slave mode. 1b - Bit clock is generated internally in Master mode.
23 BYP	Bit Clock Bypass Bypasses the bit clock divider, internal bit clock is divide by 1 of the audio master clock. 0b - Internal bit clock is generated from bit clock divider. 1b - Internal bit clock is divide by one of the audio master clock.
22-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 DIV	Bit Clock Divide Divides down the audio master clock to generate the bit clock when configured for an internal bit clock. The division value is (DIV + 1) * 2.

71.5.1.16 Receive Configuration 3 (RCR3)

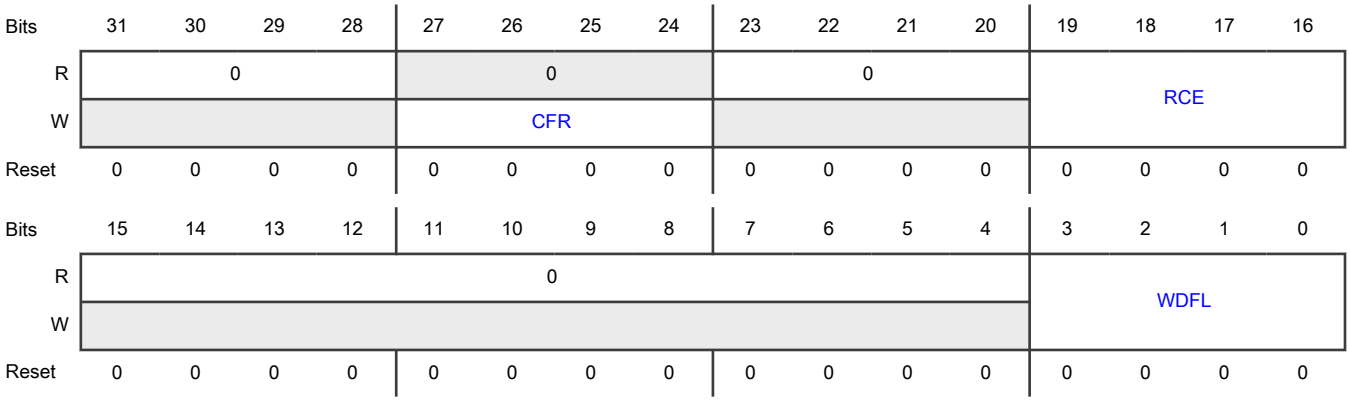
Offset

Register	Offset
RCR3	94h

Function

Contains the receive channel settings.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 CFR	Channel FIFO Reset Resets the FIFO pointers for a specific channel. Reading this field will always return zero. FIFO pointers should only be reset when a channel is disabled or the FIFO error flag is set. The width of CFR field = the number of receive channels (call it N). For example, if CFR is 2 bits wide, then bit position 24 refers to receive channel 1 FIFO pointer and bit position 25 refers to receive channel 2 FIFO pointer. Setting bit 24 resets receive channel 1 FIFO pointer, and setting bit 25 enables receive channel 2 FIFO pointer. Setting bit N will reset receive channel N FIFO pointer.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - No effect.
1b - Receive data channel N FIFO is reset.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI_0</td><td>RCR3</td><td>—</td></tr><tr><td>SAI_1</td><td>—</td><td>RCR3</td></tr></table></td></tr>
<tr><td>23-20
—</td><td>Reserved</td></tr>
<tr><td>19-16
RCE</td><td>Receive Channel Enable
Enables the corresponding data channel for receive operation. Changing this field will take effect immediately for generating the FIFO request and warning flags, but at the end of each frame for receive operation.
The width of RCE field = the number of receive channels (call it N). For example, if RCE field is 2 bits wide, then bit position 16 refers to receive channel 1 and bit position 17 refers to receive channel 2. Setting bit 16 enables receive channel 1, and setting bit 17 enables receive channel 2. Setting bit N will enable receive channel N.
NOTE
When there is only a single channel, there is no need for individual channel FIFO reset (RCR3[CFR]). In the case of a single channel, use the global FIFO reset (RCSR[FR]).
0b - Receive data channel N is disabled.
1b - Receive data channel N is enabled.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI_0</td><td>RCR3</td><td>—</td></tr><tr><td>SAI_1</td><td>RCR3[16]</td><td>RCR3[19–17]</td></tr></table></td></tr>
<tr><td>15-4</td><td>Reserved</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3-0 WDFL	Word Flag Configuration Configures which word the start of word flag is set. The value written should be one less than the word number (for example, write zero to configure for the first word in the frame). When configured to a value greater than the Frame Size field, then the start of word flag is never set.

71.5.1.17 Receive Configuration 4 (RCR4)

Offset

Register	Offset
RCR4	98h

Function

Contains the receive fields for FIFO Combine Mode, FIFO Packing Mode, and frame sync settings.

NOTE

This register must not be altered when RCSR[RE] is set.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			SYWD					0		Reserved	MF	FSE	ONDEM	FSP	FSD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28	FIFO Continue on Error

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>FCONT</td><td>Configures when the SAI will continue receiving after a FIFO error has been detected.
0b - On FIFO error, the SAI will continue from the start of the next frame after the FIFO error flag has been cleared.
1b - On FIFO error, the SAI will continue from the same word that caused the FIFO error to set after the FIFO warning flag has been cleared.</td></tr>
<tr><td>27-26
FCOMB</td><td>FIFO Combine Mode
When FIFO combine mode is enabled for FIFO reads, software reading any FIFO data register will alternate the read among the enabled data channel FIFOs. For example, if two data channels are enabled then the first read will be performed to the first enabled data channel FIFO and the second read will be performed to the second enabled data channel FIFO. Reseting the FIFO or disabling FIFO combine mode for FIFO reads will reset the pointer back to the first enabled data channel.
When FIFO combine mode is enabled for FIFO writes from the receive shift registers, the first enabled data channel input will alternate between the enabled data channel FIFOs. For example, if two data channels are enabled then the first unmasked received word will be stored in the first enabled data channel FIFO and the second unmasked received word will be stored in the second enabled data channel FIFO. Since the first word of the frame is always stored in the first enabled data channel FIFO, it is recommended that the number of unmasked words per frame is evenly divisible by the number of enabled data channels.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI_0</td><td>RCR4</td><td>—</td></tr><tr><td>SAI_1</td><td>—</td><td>RCR4</td></tr></table>00b - FIFO combine mode disabled.
01b - FIFO combine mode enabled on FIFO writes (from receive shift registers).
10b - FIFO combine mode enabled on FIFO reads (by software).
11b - FIFO combine mode enabled on FIFO writes (from receive shift registers) and reads (by software).</td></tr>
<tr><td>25-24
FPACK</td><td>FIFO Packing Mode
Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8-bit or 16-bit then only the first 8-bit or 16-bits are stored to the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO read pointer will only increment when the full 32-bit FIFO word has been read by software.
00b - FIFO packing is disabled</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Reserved. 10b - 8-bit FIFO packing is enabled 11b - 16-bit FIFO packing is enabled
23-20 —	Reserved
19-16 FRSZ	Frame Size Configures the number of words in each frame. The value written must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 16 words.
15-13 —	Reserved
12-8 SYWD	Sync Width Configures the length of the frame sync in number of bit clocks. The value written must be one less than the number of bit clocks. For example, write 0 for the frame sync to assert for one bit clock only. The sync width cannot be configured longer than the first word of the frame.
7-6 —	Reserved
5 —	Reserved Software should only write zero to this bit.
4 MF	MSB First Configures whether the LSB or the MSB is received first. 0b - LSB is received first. 1b - MSB is received first.
3 FSE	Frame Sync Early 0b - Frame sync asserts with the first bit of the frame. 1b - Frame sync asserts one bit before the first bit of the frame.
2 ONDEM	On Demand Mode When set, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is clear. 0b - Internal frame sync is generated continuously. 1b - Internal frame sync is generated when the FIFO warning flag is clear.
1	Frame Sync Polarity

Table continues on the next page...

Table continued from the previous page...

Field	Function
FSP	Configures the polarity of the frame sync. 0b - Frame sync is active high. 1b - Frame sync is active low.
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0b - Frame Sync is generated externally in Slave mode. 1b - Frame Sync is generated internally in Master mode.

71.5.1.18 Receive Configuration 5 (RCR5)

Offset

Register	Offset
RCR5	9Ch

Function
This register must not be altered when RCSR[RE] is set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			WNW					0			W0W				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31-29 —	Reserved
28-24	Word N Width

Table continues on the next page...

Table continued from the previous page...

Field	Function
WNW	Configures the number of bits in each word, for each word except the first in the frame. The value written must be one less than the number of bits per word. Word width of less than 8 bits is not supported.
23-21 —	Reserved
20-16 W0W	Word 0 Width Configures the number of bits in the first word in each frame. The value written must be one less than the number of bits in the first word. Word width of less than 8 bits is not supported if there is only one word per frame.
15-13 —	Reserved
12-8 FBT	First Bit Shifted Configures the bit index for the first bit received for each word in the frame. If configured for MSB First, the index of the next bit received is one less than the current bit received. If configured for LSB First, the index of the next bit received is one more than the current bit received. The value written must be greater than or equal to the word width when configured for MSB First. The value written must be less than or equal to 31-word width when configured for LSB First.
7-0 —	Reserved

71.5.1.19 Receive Data (RDR0 - RDR3)

Offset

Register	Offset
RDR0	A0h
RDR1	A4h
RDR2	A8h
RDR3	ACh

Function

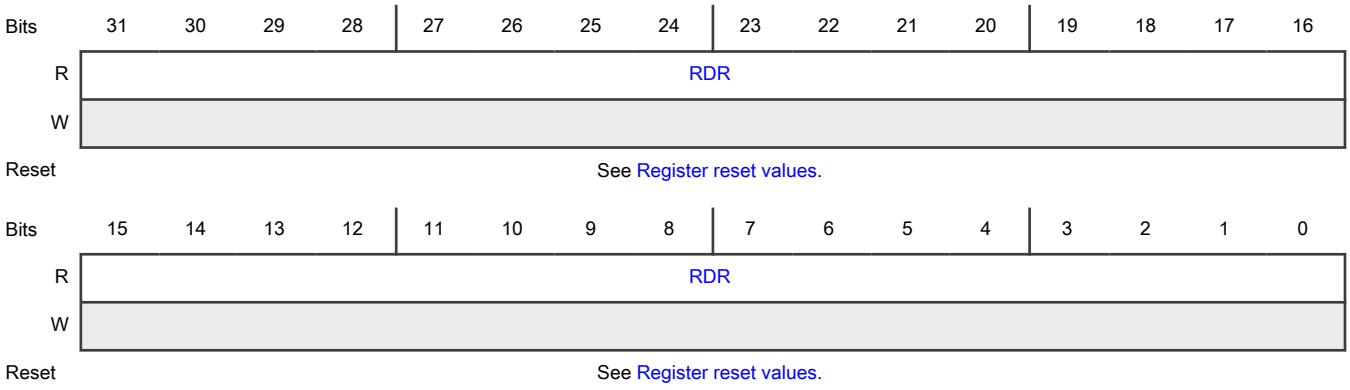
Reads from this register when the receive FIFO is not empty return the data from the top of the receive FIFO. Reads from this register when the receive FIFO is empty are ignored.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
SAI_0	RDR0–RDR3	—
SAI_1	RDR0	RDR1–RDR3

Diagram



Register reset values

Register	Reset value
RDR0	SAI_0,SAI_1: 0000_0000h
RDR1–RDR3	0000_0000h

Fields

Field	Function
31-0 RDR	Receive Data Register

71.5.1.20 Receive FIFO (RFR0 - RFR3)

Offset

Register	Offset
RFR0	C0h
RFR1	C4h
RFR2	C8h
RFR3	CCh

Function

The MSB of the read and write pointers is used to distinguish between FIFO full and empty conditions. If the read and write pointers are identical, then the FIFO is empty. If the read and write pointers are identical except for the MSB, then the FIFO is full.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
SAI_0	RFR0–RFR3	—
SAI_1	RFR0	RFR1–RFR3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												WFP			
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCP	0											RFP			
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
RFR0	SAI_0,SAI_1: 0000_0000h
RFR1–RFR3	0000_0000h

Fields

Field	Function
31-20 —	Reserved
19-16 WFP	Write FIFO Pointer FIFO write pointer for receive data channel.
15 RCP	Receive Channel Pointer When FIFO Combine mode is enabled for reads, indicates that this data channel is the next FIFO to be read.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SAI_0</td><td>RFR0–RFR3</td><td>—</td></tr>
<tr><td>SAI_1</td><td>—</td><td>RFR0</td></tr>
</table>
0b - No effect.
1b - FIFO combine is enabled for FIFO reads and this FIFO will be read on the next FIFO read.
</td></tr>
<tr><td>14-4
—</td><td>Reserved</td></tr>
<tr><td>3-0
RFP</td><td>Read FIFO Pointer
FIFO read pointer for receive data channel.</td></tr>
</table>

71.5.1.21 Receive Mask (RMR)

Offset

Register	Offset
RMR	E0h

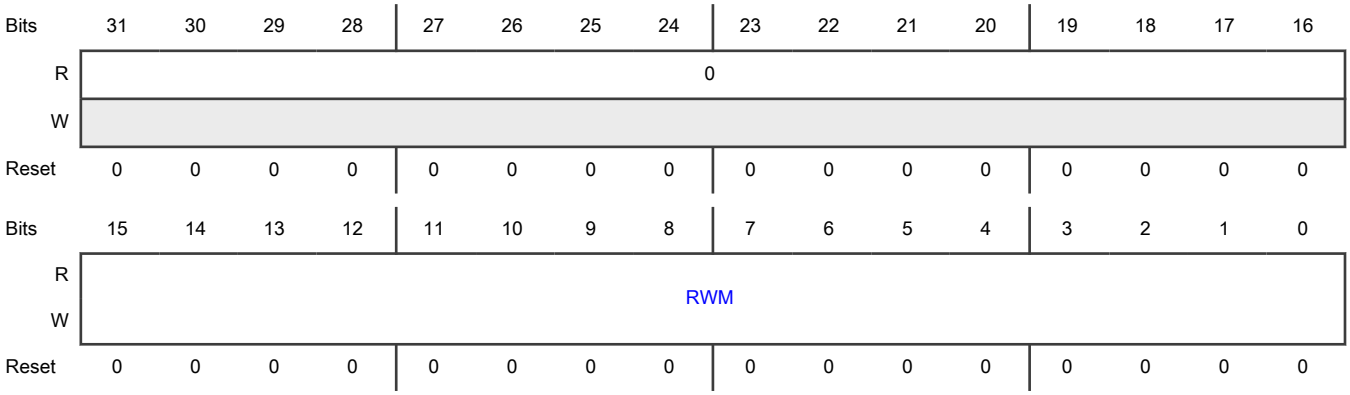
Function

This register is double-buffered and updates:

1. When RCSR[RE] is first set
2. At the end of each frame

This allows the masked words in each frame to change from frame to frame.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 RWM	Receive Word Mask Configures whether the receive word is masked (received data ignored and not written to receive FIFO) for the corresponding word in the frame. 0000_0000_0000_0000b - Word N is enabled. 0000_0000_0000_0001b - Word N is masked.

71.6 Glossary

FIFO	First-in first-out
LSB	Least significant bit
MSB	Most significant bit

Chapter 72
Ethernet Media Access Controller (EMAC)

72.1 Chip-specific EMAC information

72.1.1 EMAC instance and configuration

This chip supports up to 1 instance of EMAC IP.

Table 463. EMAC instance

Instance	MWCT2D16S/MWCT2D17S	MWCT2015S/MWCT2016S
EMAC	Yes	No

NOTE

DWC_EQOS refer to the Synopsys Ethernet MAC IP.

NOTE

The description for some EMAC registers include references to configurations of the Synopsis Ethernet MAC IP. Refer to MAC_HW_Feature registers for the specific features available in this chip.

NOTE

EMAC operates only in Clock options A and B, since the module clock becomes lower than the protocol clock (RMII/MII clocks) in other modes.

Following key features are supported:

- This chip supports only MII/RMII interfaces
- 4 bit MII interface operating at 2.5/25/50Mhz, MII can run at 10/100/200 Mbps.
- 2 bit RMII interface operating at 50Mhz, RMII can run at 10/100Mbps
- 4 bit MII-Lite interface operating at 2.5/25Mhz
- 4 PPS are supported using DMA trigger
- 200Mbps data rate is supported on Ethernet MII interface
- MTL Receive FIFO size 8192bytes
- MTL Transmit FIFO size 8192bytes

NOTE

The DMA referred within EMAC chapter refers to the internal EMAC DMA engine and not the device AXBS master DMA.

NOTE

Register access beyond 8k address space is not supported and will generate a transfer error.

NOTE

Always use store and forward mode if possible. In cut through mode, if multiple masters are active then there is chance (even in round robin arbitration mode) that EMAC DMA may starve for data resulting in insufficient data in TX FIFO leading to Tx underflow. To avoid this, increase MTL_TxQ0_Operation_Mode[TTC] value as needed.

The table below provides the information on the specific configuration utilized for the MWCT2D17S and MWCT2D16S devices.

Table 464. Specific configuration information

Application Interface		
	Application interface configuration	EQOS-AHB
	Data width	32
	Endian mode	Little Endian
	Address width	32
	CSR interfcae	APB3 interface
General feature		
	Mode of operation	10/100
	Enable double VLAN processing	Yes
	Enable Que/channel based VLAN tag inertion on Tx	Yes
	Enable SA and VLAN insertion on Tx	Enabled by default
Buffer management		
	MTL Receive FIFO size in bytes	8192
	MTL Receive FIFO size in bytes	8192
	Enable debug memory access	Yes
	Use single port memory	Yes
PHY interface		
	Enable RGMII, RMII only	Enable RMII, GMII/MII default
	Enabled signel phy interface	No
	Enable MDIO interface	Yes
Filtering		
	Enable Additional 1-31 MAC address registers	2
	Enable Address Filter Hash Table	Yes
	Hash Table Size	64
	Enable VLAN Hash table based filtering	Yes
	Extended Rx VLAN Filter Enable	Yes
	Number of VLAN tag Filters	4
	Enable Layer3 and Layer4 Packet Filter	Yes
	Number of layer3 and Layer 4 packet filters	4
	Enable Flexible Programmable Receive Parser	Yes

Table continues on the next page...

Table 464. Specific configuration information (continued)

	Maximum Packet Header Size for Parsing	128
	Maximum Entries of Parser Look up table	64
	Packet duplication support	Yes
IEEE 1588 timestamp		
	Enable IEEE 1588 Time stamp support	Enabled by default
	IEEE 1588 System Time Support	Both option
	Add IEEE 1588 Higher word register	Yes
	Enable IEEE 1588 sub nanoseconds time stamp support	Yes
	Add IEEE 1588 Auxilary snapshot	No
	Enable Flexible Pulse Per Seconds Output	Yes
	Number of pulse per Second outputs	4
	Enable PTP Timestamp offload feature	No
	Enable One Step timestamp for PTP over UDP/IP feature	No
	Enable One Step timestamp feature	Yes
Multiple queue support		
	Number of transmit Queue	2
	Number of recieve Queue	2
	Enable data center Bridging	No
	Enable Audo Vedio Bridging	Yes
	Enable support for AV in Tx queue 1	No
	Number of DMA Channel on the receive side	2
Time sensitive networking		
	Enable Enhancement to Schedule traffic(EST)	Yes
	Width of Time Interval Field in the Gate Control List	24
	Depth of Gate Control List	256
	Enable Frame Pre-emption support	Yes
Time based scheduling		
	Enable time based Scheduling	Yes
TCP/IP offloading features		

Table continues on the next page...

Table 464. Specific configuration information (continued)

	Enable receive TCP/IP Checksum Check	Enabled by default
	Enable transmit TCP/IP Checksum offload	Enabled by default
	Enable support for Tx COE in Tx queue 0	Yes
	Enable support for Tx COE in Tx queue 1	No
	Enable TCP Segmentation offloading for TCP/IP	No
	Enable Split Header Feature	No
	Enable IPv4 ARP offload	No
RMON counters		
	Enable MAC Management Counters(MMC)	Yes
	Enable the MAC Management Counters for Reciever TCP/IP Checksum offload	No
Automotive safety feature		
	Enable all automotive safety feature	Yes

72.1.2 Impact of MAC flush commands on MAC_PPSx_Target_Time_Seconds register

If the software discards the media clock generation request by configuring the PPSCTRL_PPSCMD (or PPSCMD#i) field of MAC_PPS_Control register to 0, the MAC does not flush the captured presentation time, i.e., the MAC_PPSx_Target_Time_Seconds register contents don't get changed.

72.3 Introduction

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

EMAC:

- Supports the 10/100 Mbps application in compliance with IEEE802.3-2015 specifications
- Can support advanced applications such as time-sensitive networking and AVB
- Supports a MAC core that has prominent features such as a flexible receive parser, media clock recovery and generation, and safety

72.3.1 Features

EMAC supports these features in addition to the default feature defined in the IEEE802.3 specification:

- MII (10/100 Mbps), RMII (10/100 Mbps)
- Time-aware shaper (IEEE802.1bv), time synchronization (IEEE802.1AS-rev), and frame preemption (IEEE802.1Qbu) for time-sensitive networking
- Media clock recovery and generation for AVB
- AMBA 2.0 for AHB master and APB3 interface
- RMII specification version 1.2 from RMII consortium
- Separate interface for transmit, receive, and control paths
- Two-buffer ring

- Broadcast and multicast packet duplication
- Full-duplex flow control operations (IEEE 802.3x pause packet and priority flow control)
- Network statistics with MAC management (RMON) counters
- Flexibility to control the pulse per second (PPS) output signal
- MDIO clause 22 and clause 45 interface for configuration and management of the PHY device
- Preamble and SFD insertion and deletion
- Automatic CRC and pad generation/stripping option
- Option to disable CRC checking
- Programmable insert packet gap
- Source address insertion or replacement
- VLAN insertion, replacement, and deletion in transmitted packets with per-packet or static-global control, insertion, replacement, and deletion of up to two VLAN tags, insertion, replacement, and deletion of queue or channel-based VLAN tags
- IEEE802.1Q VLAN tag detection and an option to delete the tags in the receive packets
- Programmable safety watchdog timeout limits
- Flexible address filtering modes that allow:
  - Up to two additional 48-bit perfect DA filters with masks for each byte
  - Up to two 48-bit SA comparison checks with masks for each byte
  - 64-bit hash filter for multicast and unicast DA addresses
  - Option to pass all multi-cast addressed packets
  - Promiscuous mode to pass all packets without any filtering for network monitoring
  - Passing of all incoming packets (according to filter) with a status report
- Additional packet filtering that is based on:
  - Virtual local area network (VLAN) tag—perfect match and hash-based filtering, which is based on either outer or inner VLAN tag, is possible
  - Layer 3 and layer 4—TCP or UDP over IPv4 or IPv6
  - Extended VLAN tag—filtering based on four filter selections

NOTE

The DCB feature discussed in the following section(s) is not supported.

72.3.2 Functional block diagram

This figure shows the EMAC block diagram. EMAC has four main blocks to perform all the functions.

- The AHB interface is connected to all DMA channels.
- The DMA arbiter helps in the arbitration of all paths (transmit and receive) in DMA channels.
- Each channel has a separate set of control and status registers (CSRs) for managing the transmit and receive functions, descriptor handling, and interrupt handling.

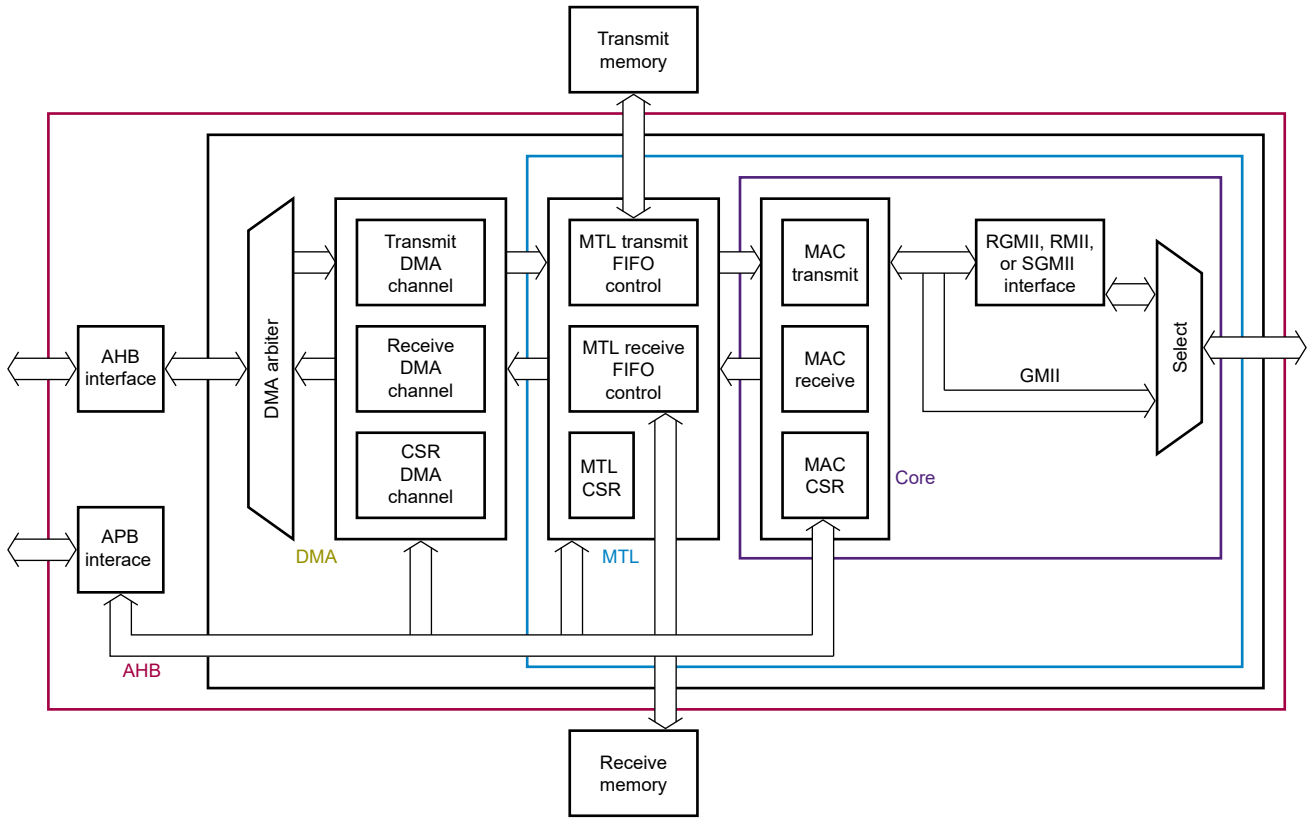


Figure 390. System-level block diagram

72.4 Architecture

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

The section describes EMAC interfaces, protocols, and functionality. These are its main blocks:

- AHB master and APB3 slave interface
- DMA controller
- MTL
- MAC
- Interrupts

72.4.1 AHB master and CSR slave (APB3) interface

The 32-bit APB3 slave interface:

- Provides access to DMA, MTL, and MAC control and status registers (CSRs)
- Supports the 32-bit write and read transfers to these CSRs
- Supports single- and all-burst transfers, and an OKAY response

The DMA block uses the AHB master interface to interact with the external world. The AHB master interface:

- Controls data transfer
- Is AMBA 2.0-compliant with no restrictions
- Supports 32-bit data transfer

- Supports single and INCR transfers
- Supports burst-length modes such as Fixed, Unspecified, and Mixed

72.4.2 DMA inclusions

DMA controller:

- Includes independent transmit (Tx) and receive (Rx) engines, and a CSR space. The transmit engine transfers data from the system memory to the MTL interface, whereas the receive engine transfers data from the MTL interface to the system memory.
- Is designed for packet-oriented data transfers such as packets in Ethernet.
- Communicates with the host through CSR, descriptor lists, and data buffers.

DMA descriptors

DMA supports up to two transmit and two receive descriptor lists (or DMA channels). The base address of each list is written to the respective transmit and receive descriptor list address registers. A descriptor list is forward linked and the next descriptor is always considered at a fixed offset to the current one, and DMA_CH0_Control[DSL] contols the offset. The number of descriptors in the list is programmed using the respective DMA Channel 0 Tx Descriptor Ring Length (DMA_CH0_TxDesc_Ring_Length) and DMA Channel 0 Rx Descriptor Ring Length (DMA_CH0_RxDesc_Ring_Length) registers. After the DMA processes the last descriptor in the list, it automatically jumps to the descriptor in the list address register to create a descriptor ring.

EMAC supports the ring structure for DMA descriptors. For more information, see Descriptors .

The DMA engine uses descriptors to efficiently move data from source to destination with minimal host intervention, and the DMA controller can be programmed to interrupt the host in certain situations. These situations may include packet transmit and receive transfer completion and other normal or error conditions.

The descriptor lists reside in the physical memory address space of the application. Each descriptor can point to a maximum of two buffers in the system memory, enabling two buffers to be used and physically addressed rather than contiguous buffers in the memory.

Data buffers

A data buffer resides in the application physical memory space and consists of an entire packet or part of a packet but cannot exceed a single packet. Buffers contain only data. The buffer status is maintained in the descriptor. Data chaining refers to packets that span multiple data buffers. However, a single descriptor cannot span multiple packets. The DMA skips to the data buffer of the next packet when an EOP is detected.

The next few sections discuss DMA controller.

72.4.2.1 DMA controller bus burst access

DMA engines attempt to transfer data in a burst of the maximum size programmed using DMA_CH0_Tx_Control[TxPBL] and DMA_CH0_Rx_Control[RxPBL] of the respective DMAs, which always access the receive and transmit descriptors in the maximum possible (limited by PBL or 16 * 8/bus width) burst length for 16 bytes to be read. The burst transfers that the DMA initates can be split into multiple burst transfers according to the AHB requirements and settings specified in DMA System Bus Mode (DMA_SysBus_Mode).

The transmit DMA initiates a data transfer only when sufficient space is available in the MTL transmit queue to accommodate either of these:

- Bytes corresponding to the configured burst (PBL * bus_width/8)
- Remaining bytes in the transmit buffer without EOP
- Number of bytes till EOP

The receive DMA initiates a data transfer in these conditions:

- Sufficient data is available in the MTL receive queue to accommodate the configured burst
- EOP (when it is less than the configured burst length) is detected in the receive queue

DMA indicates the start address and the number of transfers required to the AHB master interface. When the interface is configured for fixed-length burst, it transfers data by using the best combination of the INCR4, INCR8, or INCR16 and SINGLE transactions. If EOP is reached before the fixed burst ends on the AHB interface, DMA performs dummy transfers to complete the fixed burst. Otherwise, DMA_SysBus_Mode[FB] becomes 0. The DMA transfers the data using undefined length (INCR) and SINGLE transactions.

When the AHB interface is configured for address-aligned beats, both DMA engines ensure that the first-burst transfer that AHB initates is less than or equal to the size of the configured PBL value, and the subsequent beats start at an address aligned to this value. The DMA can only align the address for beats up to size 16 (for PBL > 16) for the AHB interface because it does not support more than INCR16.

72.4.2.2 DMA application data buffer alignment

The transmit and receive data buffers do not have any restrictions on start address alignment. For example, in systems with 32-bit memory, the start address for buffers can be aligned to any of the four bytes. However, the DMA always initiates write transfers with an address aligned to the bus width and dummy data (old data) in the invalid byte lanes. The process of write transfer typically happens during the beginning or end of an Ethernet packet transfer. The software driver must discard the dummy bytes based on the start address of the buffer and size of the packet.

Table 465. Data buffer alignment examples

Examples	
Buffer read	If the transmit buffer address is FF2h (for a 32-bit data bus) and 15 bytes need to be transferred, the DMA reads five full words from address FF0h. However, when DMA transfers data to the MTL transmit queue, the extra bytes (the first two bytes) are dropped or ignored. Similarly, the last 3 bytes of the last transfer are also ignored. The DMA always ensures that it transfers a full 32-bit data to the MTL transmit queue, unless it is the end of the packet.
Buffer write	If the receive buffer address is FF2h (for a 64-bit data bus) and 16 bytes of a received packet need to be transferred, the DMA writes 3 full words from address FF0h. However, the first 2 bytes of the first transfer and the last 6 bytes of the third transfer have dummy data. The DMA considers the offset address only if it is the first receive buffer of the packet. It ignores the offset address and performs full-word writes for the middle and last receive buffers of the packet.

72.4.2.3 DMA buffer size calculations

DMA does not update the size fields in the transmit and receive descriptors. It updates only the status fields (RDES and TDES) of the descriptors. The driver performs the size calculations.

The transmit DMA transfers the exact number of bytes (as the buffer size field of TDES2 indicates) to MAC. If a descriptor is marked as first (TDES3[FD] = 1), the DMA marks the first transfer from the buffer as SOP, and if the descriptor is marked as last (TDES3[LD]), the DMA marks the last transfer from that data buffer as EOP to the MTL.

The receive DMA transfers data to a buffer until the buffer is full or MTL sends the end of packet. When TDES3[FD] = 1, the amount of valid data in a buffer is accurately indicated by the buffer size fields in DMA Channel Rx Control (DMA_CH0_Rx_Control) minus the data buffer pointer offset. The offset is 0 when the data buffer pointer is aligned to the data bus width. If a descriptor is marked as last, the buffer may not be full (as indicated by DMA_CH0_Rx_Control[RBSZ_13_y] and DMA_CH0_Rx_Control[RBSZ_x_0]). To compute the amount of valid data in this final buffer, the driver must read the packet length (PL fields of RDES3[14:0]) and subtract the sum of the buffer sizes of the preceding buffers in this packet. The receive DMA always transfers the start of the next packet with a new descriptor.

72.4.2.4 DMA arbiter

The arbiter inside the DMA module performs the arbitration between the transmit and receive channel accesses to the AHB master interface. These two types of arbitrations are supported:

- Round-robin: If DMA_Mode[DA] = 0 and both transmit and receive DMAs simultaneously request access, the arbiter allocates the data bus in ratio sets defined in DMA_Mode[PR].
- Fixed-priority: If DMA_Mode[DA] = 1, receive DMA is always prioritized over transmit DMA for data access by default. If DMA_Mode[TXPR] = 1, transmit DMA is prioritized over receive DMA as explained in Table 482.

72.4.3 MTL

MTL provides the FIFO memory interface to buffer and regulate the packets between system memory and MAC. It also enables data to be transferred between the system clock and MAC clock domains. The MTL layer has two data paths: transmit path and receive path. MTL communicates with the host through ATI on the transmit path and through ARI on the receive path.

72.4.3.1 Transmit path

The internal DMA:

- Handles all transactions for the transmit path through ATI.
- Pushes the Ethernet packet reads from system memory to the corresponding queue.

The Ethernet packet, then, pops out and is transfered to MAC when the queue reaches its threshold (Threshold mode) or if the complete packet is in the queue (Store-and-Forward mode). When EOP is transferred, the status of the transmission is taken from MAC and transferred back to the internal DMA.

72.4.3.1.1 Transmit control word

This control information related to packet transmission is provided as part of the control word through the ATI interface:

- Packet length (if DCB is enabled with WFQ scheduling algorithm)
- CRC pad control
- Source address insertion control
- VLAN insertion and replacement control and VLAN tag for outer and inner VLAN tags
- TCP/IP checksum insertion control
- One-step timestamping control correction
- Transmit timestamp enable

72.4.3.1.2 Transmit operation

These two modes of operation trigger data read towards MAC:

- Threshold (or cut-through) mode: This is the default mode. In this mode, as soon as the number of bytes in the queue crosses the configured threshold level (or when the end of packet is written before the threshold is crossed), the data is ready to be popped out and forwarded to MAC. The threshold level is configured by using MTL_TxQ0_Operation_Mode[TTC].
- Store-and-Forward mode: In this mode, MTL pops the packet out to MAC only when one or more of the following conditions are true:
  - A complete packet is stored in the queue.
  - The transmit FIFO becomes almost full.
  - The ATI watermark becomes low.

  The watermark becomes low when the requested queue is not spacious enough to accommodate the requested burst length on ATI. Therefore, MTL, when operating in Store-and-Forward mode, allows packet transmission even if the packet length is bigger than the transmit queue size.

You can flush the complete content of the transmit queue by writing 1 to MTL_TxQ0_Operation_Mode[FTQ] or MTL_TxQ1_Operation_Mode[FTQ], depending on the queue. Doing so initializes the queue pointers to the default state. The FTQ

field returns to 0 by itself. If you write 1 to the FTQ field during a packet transfer from MTL to MAC, MTL stops further transfers because MTL considers the queue to be empty. Therefore, an underflow event occurs at the MAC transmitter.

72.4.3.1.3 Initialization flow

After reset, MTL is ready to manage the data flow between DMA and MAC.

- With single-transmit queue configurations, there are no initialization requirements for enabling MTL.
- With multiple-transmit queue configurations, initialize the queue size for each of the queues by programming MTL_TxQ0_Operation_Mode[TQS] corresponding to a transmit queue. Also, initialize the MAC block. Internal DMA controllers must be individually enabled through their respective CSRs.

72.4.3.2 Receive path

MAC sends packets to the MTL receive module and pushes them into the receive queue. MTL indicates the status (fill level) of the queue to the application or DMA when it crosses the configured receive threshold (MTL_RxQ0_Operation_Mode[RTC]), or when the MTL receive module receives the complete packet in Store-and-Forward mode. MTL also indicates the fill level of the queue so that DMA can initiate preconfigured burst transfers to the AHB interface.

72.4.3.2.1 Receive operation

During a receive operation, MTL is MAC's slave. This is the general sequence of events:

1. When MAC receives a packet, it indicates the availability of receive data.
2. MAC indicates the SOP and EOP delimiters.
3. MTL accepts the data and pushes it into the corresponding receive queue.
4. When MAC transfers EOP to the MTL receive module, MAC also drives the status word that MTL pushes into the corresponding receive queue.
5. MTL takes the data out of the queue and sends it to DMA depending on these modes:
   - Threshold mode
   - Store-and-Forward mode

   The sections that follow discuss these modes.

72.4.3.2.1.1 Threshold mode

In Threshold (default) mode, MTL reads the data and indicates its availability to the application or DMA when one of these occurs:

- Data bytes equal to the threshold amount are written to the receive queue (to MTL_RxQ0_Operation_Mode[RTC] and MTL_RxQ1_Operation_Mode[RTC]).
- A full packet of data is received into the queue.

72.4.3.2.1.2 Store-and-Forward mode

In this mode (when MTL_RxQ0_Operation_Mode[RSF] = 1), the initial receive queue locations are reserved for the status words before writing the SOP. A packet is read out only after it is completely written into the receive queue. In this mode, all error packets are dropped (if configured through MTL_RxQ0_Operation_Mode[FEP]) in such a way that only valid packets are read and forwarded to the application.

72.4.3.2.2 Multi-packet receive operation

In Threshold mode, the packet status is available immediately after the packet data. In Store-and-Forward mode, the packet data is available after the packet status. MTL is capable of storing any number of packets in the queue as long as it is not full.

If MAC receives a packet when the corresponding receive queue is full, MTL ignores that packet and overflow pulse generates on the corresponding receive queue. In addition, MTL increments the overflow counter in MTL Rx Queue Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt) for the corresponding queue.

72.4.3.2.3 Error handling in receive operation

MTL performs these actions if the MTL receive queue is full before it receives the EOP data from MAC:

1. Declares an overflow
2. Drops the whole packet (including the status word)
3. Increments the overflow counter in DMA (MTL Rx Queue Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt))

This is true even if MTL_RxQ0_Operation_Mode[FEP] is 1.

If the start address of such a packet is already transferred to the read controller, the rest of the packet is dropped and a dummy EOP is written to the queue along with the status word with overflow status. The status indicates a partial packet because of overflow. In such packets, the Packet Length field is invalid. If the MTL receive queue is configured to operate in the Store-and-Forward mode and the length of the received packet is more than the queue size, overflow occurs and all such packets are dropped.

The MTL receive control logic can filter errors and undersized packets, if enabled by using MTL_RxQ0_Operation_Mode[FEP] and MTL_RxQ0_Operation_Mode[FUP]. If the start address of one such packet is already transferred to the receive queue read controller, the packet is not filtered. The start address of the packet is transferred to the read controller after the packet crosses the receive threshold defined using MTL_RxQ0_Operation_Mode[RTC].

If the MTL receive queue is configured to operate in Store-and-Forward mode, all error packets can be filtered and dropped. For the application or DMA to flush the error packet being read from the queue, it must assert the flush signal. MTL then stops transferring data to the application (DMA). It internally reads the rest of the packet and drops it. MTL then starts transferring the next packet, if it's available.

72.4.4 MAC

MAC can support the MII and RMII PHY interface and it consists of:

- MTI
- MRI
- MCI

72.4.4.1 MAC transmission

The MAC transmission process is as follows:

1. The transmission initiates when MTL pushes in data with the SOP signal asserted.
2. After the SOP signal is detected, MAC accepts the data and begins transmitting to RMII or MII.
3. After the EOP signal is transferred to MAC, it performs one of these steps:
   - Completes normal packet transmission and sends the transmission status to MTL
   - Sends retry requests if a normal collision (in Half-Duplex mode) occurs during transmission, and until one of the following is true:
     - Packet is successfully transmitted.
     - Maximum retry requests have expired. When this happens, MAC aborts the packet transmission with "excessive collision transmit" status. MAC accepts and drops all further data until it receives the next SOP. The MTL block must retransmit the same packet from SOP on observing a retry request (in the status) from MAC.
   - If any one of the following happens, MAC aborts the packet transmission:

- No carrier (Half-Duplex mode)
- Loss of carrier (Half-Duplex mode)
- Excessive deferral (Half-Duplex mode)
- Late collisions (Half-Duplex mode)
- Jabber
- MAC accepts and drops all further data until it receives the next SOP

4. MAC issues an underflow status if MTL is not able to provide data continuously during transmission. Until the next SOP is received, MAC accepts and drops all further data.
5. During the normal transfer of a packet from MTL, if MAC receives an SOP without getting an EOP for the previous packet, it ignores the SOP and considers the new packet as a continuation of the previous packet.

72.4.4.2 MAC reception

The receive operation initiates as follows:

1. MAC detects a state-of-frame data on RMII or MII.
2. MAC strips the preamble and SFD before processing an Ethernet packet.
3. The MAC AFM:
   - Checks the header fields (SA and DA) of an incoming packet for filtering
   - Verifies the CRC contained in the packet's FCS field
4. MAC's AFM checks the header fields (SA and DA) of an incoming packet for filtering.
5. FCS verifies the CRC of the received packet.
6. MAC stores the received packet in a shallow buffer until address filtering is performed.
7. AFM drops the packet if it fails the address filter.

72.4.5 Interrupts

EMAC supports interrupt coalescing, which reduces the number of interrupts generated by the module and reduces the CPU load. MAC Interrupt Status (MAC_Interrupt_Status) captures various interrupt events. If the interrupt enable field of an interrupt is 1, the corresponding interrupt generates based on the event status in the Status registers. The Interrupt mode (INTM) field decides whether the interrupt signal is a level signal or pulse signal. See the following table and interrupt map file attached to this document for details.

Table 466. Interrupt request

Interrupt request	Interrupt Enable / Mask Register	Interrupt Flag Register
Common interrupt	MAC Interrupt Enable (MAC_Interrupt_Enable)	MAC Interrupt Status (MAC_Interrupt_Status)
	MMC Receive Interrupt Mask (MMC_Rx_Interrupt_Mask)	MMC Receive Interrupt (MMC_Rx_Interrupt)
	MMC Transmit Interrupt Mask (MMC_Tx_Interrupt_Mask)	MMC Transmit Interrupt (MMC_Tx_Interrupt)
	MMC FPE Transmit Interrupt Mask (MMC_FPE_Tx_Interrupt_Mask)	MMC Transmit FPE Fragment Counter Interrupt Status (MMC_FPE_Tx_Interrupt)

Table continues on the next page...

Table 466. Interrupt request (continued)

Interrupt request	Interrupt Enable / Mask Register	Interrupt Flag Register
	MMC FPE Receive Interrupt Mask (MMC_FPE_Rx_Interrupt_Mask)	MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt)
	MTL Debug Control (MTL_DBG_CTL)	MTL Debug Status (MTL_DBG_STS)
	MTL EST Interrupt Enable (MTL_EST_Intr_Enable)	MTL EST Status (MTL_EST_Status)
	MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status)	MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status)
	MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status)	MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status)
	MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status)	MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status)
	DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)(transfer complete interrupts only)	DMA_CH0_Interrupt_Status
	DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)(transfer complete interrupts only)	DMA_CH1_Interrupt_Status
		DMA Interrupt Status (DMA_Interrupt_Status) (First level status. Indicates major block event source: DMA channel, MTL or MAC)
		MTL Interrupt Status (MTL_Interrupt_Status) (Status summary for MTL block interrupts: Rx Parser, EST, FIFO Debug, Queue 1/Queue 0)
Tx interrupt 0/1	DMA Channel Tx Control (DMA_CH0_Tx_Control)	DMA Channel 0 Status (DMA_CH0_Status)
	DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)	
	DMA Channel 1 Tx Control (DMA_CH1_Tx_Control)	DMA Channel 1 Status (DMA_CH1_Status)
	DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)	
Rx interrupt 0/1	DMA Channel Rx Control (DMA_CH0_Rx_Control)	DMA Channel 0 Status (DMA_CH0_Status)
	DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)	
	DMA Channel 1 Rx Control (DMA_CH1_Rx_Control)	DMA Channel 1 Status (DMA_CH1_Status)
	DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)	

Table continues on the next page...

Table 466. Interrupt request (continued)

Interrupt request	Interrupt Enable / Mask Register	Interrupt Flag Register
Safety interrupt	MTL ECC Interrupt Enable (MTL_ECC_Interrupt_Enable)	MTL ECC Interrupt Status (MTL_ECC_Interrupt_Status)
		MTL Safety Interript Status (MTL_Safety_Interrupt_Status)
	MTL DPP Control (MTL_DPP_Control)	MAC DPP FSM Interrupt Status (MAC_DPP_FSM_Interrupt_Status)
	MAC FSM Control (MAC_FSM_Control)	

These are the interrupt output signals that are synchronous to the CSR clock.

- Sbd_intr_o – common interrupt
- Sbd_perch_tx_intr_o[max_dma_ch] – interrupt per transmit channel
- Sbd_perch_rx_intr_o [max_dma_ch] – interrupt per receive channel

NOTE

max_dma_ch = number of transmit/receive queues, which is 2

The sbd_intr_o common interrupt is a level signal. When it is asserted, a corresponding interrupt event source can be found in DMA Interrupt Status (DMA_Interrupt_Status), which is a read-only register that contains the event source fields corresponding to each DMA channel (transmit and receive queue pair), MAC transaction layer, and MAC blocks. You must then read the following registers and look for the fields that are 1:

- MAC Interrupt Status (MAC_Interrupt_Status)
- MTL Interrupt Status (MTL_Interrupt_Status)
- DMA Channel 0 Status (DMA_CH0_Status)

The sbd_intr_o interrupt deasserts only when all the enabled interrupt events are clear in their respective status registers, and correspondingly, all the fields in DMA Interrupt Status (DMA_Interrupt_Status) are 0. DMA Channel 0 Status (DMA_CH0_Status) captures all the interrupt events of that transmit DMA and receive DMA channel pair.

DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable) contains the corresponding enable fields for each of the interrupt events. These are the two groups of interrupts in the DMA channel:

- Normal—DMA_CH0_Status[NIS] indicates this interrupt, which is used for events that occur during the normal transfer of packets (TI, RI, TBU).
- Abnormal—DMA_CH0_Status[AIS] indicates this interrupt, which is used for error events.

Interrupts are not queued. If the same interrupt event occurs again before the driver responds to the previous one, no additional interrupts generate.

The common sbd_intr_o output signal asserts for the transfer complete interrupts only when the corresponding interrupts are enabled in DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable).

EMAC also supports these per-channel transfer-complete interrupt signals:

- sbd_perch_tx_intr_o[max_dma_ch] (transmit per channel interrupts)
- sbd_perch_rx_intr_o[max_dma_ch] (receive per channel interrupts)

The behavior of the RI/TI/sbd_perch_tx_intr_o[]/sbd_perch_rx_intr_o[] changes depends on the configuration specified in DMA_Mode[INTM]. This table explains the transfer complete interrupt behavior.

Table 467. Transfer complete interrupt behavior

Interrupt mode	Behavior of sbd_perch_tx_intr_o[] and sbd_perch_rx_intr_o[]	Behavior of the TI/RI interrupts and sbd_intr_o
INTM = 1	In this mode, these signals indicate the values of the corresponding DMA_CH0_Status[RI] and DMA_CH0_Status[TI] fields when DMA_CH0_Status[RI] and DMA_CH0_Status[TI] are 1, respectively. Therefore, they are level signals that you clear by writing 1 to these fields. The signals do not assert when DMA_CH0_Status[RI] or DMA_CH0_Status[TI] is 0.	DMA_CH0_Status[RI] and DMA_CH0_Status[TI] are configured as explained. For any RI/TI events: • The sbd_intr_o signal does not assert. • DMA_CH0_Status[NIS] remains 0.
INTM = 2	In this mode, RI/TI are queued and indicate the values of the corresponding DMA_CH0_Status[RI] and DMA_CH0_Status[TI] fields when any of these fields is 1. RI and TI are level signals that you clear by writing 1 to these fields. However, the fields become 1 again if another TI/RI event is detected before DMA_CH0_Status[RI] and DMA_CH0_Status[TI] become 1 for the previous event.	DMA_CH0_Status[RI] and DMA_CH0_Status[TI] become 1 whenever the transfer complete event is detected and are reset whenever the software driver changes them to 0 by writing 1. However, if another transfer complete event is detected before the software changes it to 0 (services it), EMAC automatically writes 1 to these fields again. The generation of the sbd_intr_o signal; however, is not based on DMA_CH0_Status[RI] or DMA_CH0_Status[TI].

72.5 Signal descriptions

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

72.5.1 Module signals

This table provides port/signal names and their descriptions. See the IOMUX file attached to this document for details.

Table 468. Module signals

Port name (clock/signal)	I/O	Description
MII_RMII_TXCLK (Clock)	I	MII: The external PHY provides this transmission clock, which operates at a frequency of 25 MHz in 100 Mbps mode and at 2.5 MHz in 10 Mbps mode. All transmission signals that MAC generates are synchronous to this clock, which is required for all PHY interfaces.

Table continues on the next page...

Table 468. Module signals (continued)

Port name (clock/signal)	I/O	Description
		RMII: The RMII interface uses this 50 MHz clock. If you select RMII mode, MII_RX_CLK (25 MHz or 2.5 MHz) must be derived from the RMII reference clock.
MII_RX_CLK (Clock)	I	The external PHY provides this receive clock for the MII and RMII interfaces. The clock operates at a frequency of 25 MHz in 100 Mbps mode and at 2.5 MHz in 10 Mbps mode. All MII receive signals that MAC receives are synchronous to MII_RX_CLK. The clock's input is required for all PHY interfaces.
EMAC_PPS[3:0] (Signals)	I/O	This group of signals is used as pulse per second in Output mode and as media clock generation trigger in Input mode. The signals trigger input to DUT to capture presentation time. Based on the presentation control value of MAC PPS Control (MAC_PPS_Control), the signals can be defined as pulse or level signals.
MII_RMII_TX_EN (Signal)	O	MAC drives this signal, which performs multiple functions depending on the selected PHY interface as described in this list: • MII: When high, indicates that valid data is being transmitted to the phy_txd_o bus. MII_RMII_TX_EN is synchronous to MII_RMII_TX_CLK. • RMII: When high, indicates that valid data is being transmitted to the phy_txd_o bus. MII_RMII_TX_EN is synchronous to MII_RMII_RX_CLK.
MII_RMII_TXD[3:0] (Signals)	O	This is a group of transmit data signals driven by MAC. These signals perform multiple functions depending on the selected PHY interface as described in this list: • MII: Bits[3:0] provide the MII transmit data nibble. The data is valid only when the MII_RMII_TX_EN signal is high. MII_RMII_TXD[3:0] is synchronous to MII_MII_TX_CLK. • RMII: Bits[1:0] provide the RMII transmit data. The data is valid

Table continues on the next page...

Table 468. Module signals (continued)

Port name (clock/signal)	I/O	Description
		only when the MII_RMII_TX_EN signal is high. MII_RMII_TXD[3:0] is synchronous to MII_RMII_TX_CLK.
MII_CRS (Signal)	I	This signal is valid only in MII mode. The PHY drives this signal high when the transmit or receive medium is not idle, and drives the signal low when both these mediums are idle. The signal is not synchronous to any clock.
MII_COL (Signal)	I	This signal is valid only in MII mode. The PHY drives this signal high when a collision is detected on the medium. The signal is not synchronous to any clock.
MII_RMII_RX_DV (Signal)	I	The PHY drives this signal. It performs multiple functions depending on the selected PHY interface as described in this list: • MII: Indicates that the data on the MII_RXD bus is valid. It remains high continuously from the first recovered byte or nibble of the packet through the final recovered byte or nibble of the packet. MII_RMII_RX_DV is synchronous to MII_RX_CLK. • RMII: Contains the CRS and data valid information of the receive interface. MII_RMII_RX_DV is synchronous to MII_RMII_TX_CLK.
MII_RMII_RX_ER (Signal)	I	The PHY drives this signal. It performs multiple functions depending on the selected PHY interface as described in this list: • MII: Indicates an error or carrier extension in the received packet of the MII_RXD[3:0] bus. MII_RMII_RX_ER is synchronous to MII_RX_CLK. • RMII: Is not used.
MII_RMII_RXD[3:0] (Signals)	I	This is a group of data signals received from the PHY. These signals perform multiple functions depending on the selected PHY interface as described in this list:

Table continues on the next page...

Table 468. Module signals (continued)

Port name (clock/signal)	I/O	Description
		• MII: Bits [3:0] provide the MII receive data nibble. The data is valid only when the MII_RMII_RX_DV signals are high. MII_RMII_RXD[3:0] is synchronous to MII_RX_CLK. • RMII: Bits [1:0] provide the RMII receive data. The data is valid only when the MII_RMII_RX_DV signal is high. MII_RMII_RXD[3:0] is synchronous to MII_RMII_TX_CLK.
MII_RMII_MDC	O	MAC provides timing reference for MII_RMII_MDIO or MII through this periodic clock. The application clock generates this clock through a clock divider that MAC_MDIO_Address[CR] controls.
MII_RMII_MDIO (Signal)	I/O	MDIO uses this signal to transfer control and data information to PHY.

72.6 Using PHY interfaces

This section is Synopsys Proprietary. Used with permission.

This module support the following modes:

- RMII 10/100 Mbps interface
- MII 10/100 Mbps interface

Phy_intf_sel input signal decides which mode is selected. Samples the Phy_intf_sel signal at reset. See Signal descriptions for more information. MAC_Configuration[PS] and MAC_Configuration[FES] selects the mode's speed.

The module supports the IEEE802.3-2015 specification for MII and RMII specification version 1.2 from RMII consortium.

NOTE

RMII reference clock (50 MHz) can be fed to IP from an external source or internally from PLL on SoC.

The module supports the access of the phy registers through SMA. It is a two wire Station management interface (MIM):

1. MDC
2. MDIO

According to IEEE 802.3 specification, maximum operating frequency of MDC is 2.5 Mhz which system clock derives using a divider. MAC_MDIO_Address[CR] programs to generate different MDC clock frequency.

SMA supports MDIO clause 45 and clause 22 frame structure per the IEEE802.3 specification. Writing 1 to MAC_MDIO_Address[C45E] enables the clause 45 frame structure.

The MII interface reduces the accuracy of the 1588 timestamp if it is overclocked to run at 200 Mbps. This happens because the MAC logic uses the MII interface speed,to adjust or compensate for the timestamps taken at MII, as compared to the time generated in the PTP clock domain as specified in MAC_Configuration[FES].

72.7 VLAN and double VLAN insertion, deletion, replacement and tagging

This section and all its sub-sections are Synopsys proprietary. These are used with permission.

The following sections describe the VLAN and double VLAN features.

72.7.1 Double VLAN processing

In the double VLAN tagging processing, the module supports the processing of two VLAN tags. The two VLAN tags are:

1. Inner VLAN tag (C-VLAN)
2. Outer VLAN tag (S-VLAN)

If there is only one tag in a packet then it is considered as an outer tag.

The module uses MAC_VLAN_Incl and MAC_Inner_VLAN_Incl and MAC_VLAN_Tag registers to support the following functions in a double VLAN processing feature.

- Insertion, replacement, or deletion of up to two VLAN tags in the transmit path.
- Packet filtering and stripping on the basis of any one of the two VLAN tags in the receive path. Stripping and providing up to two VLAN tags in the receive path as a part of the receive status.

72.7.1.1 Transmit path

Table 469. Double VLAN processing features in transmit path

Feature	Description
Support for C-VLAN and S-VLAN tag types	The inner or outer VLAN tag can be of C-VLAN and S-VLAN type. MAC_VLAN_Incl[CSVL] and MAC_Inner_VLAN_Incl[CSVL] specifies the VLAN type. The module supports the processing of any sequence of outer and inner VLAN tags. NOTE The module does not support the C-VLAN and S-VLAN sequence. MAC does not examine whether the packet, which the application provides has a valid sequence of the VLAN tag types or the insertion or replacement operation results in an invalid sequence of VLAN tag type. Therefore, the application must provide a correct sequence of VLAN tag types and program the MAC in such a way that it results in correct sequence of VLAN tag types in the transmitted packet. The application must ensure the following: • The inner tag should not be S-VLAN when outer C-VLAN Tag insertion is enabled. • The outer tag should not be C-VLAN when inner S-VLAN Tag insertion is enabled. • The inner tag should not be S-VLAN when C-VLAN replaces the outer tag. • The outer tag should not be C-VLAN when S-VLAN replaces the inner tag.
VLAN tag deletion	MAC_VLAN_Incl[VLC] or MAC_Inner_VLAN_Incl[VLC] respectively enables the VLAN tag deletion for an outer or inner tag. When you enable the VLAN deletion, MAC deletes the tag present at the corresponding position. When a packet has only one tag, it is considered as the outer tag. If you enable the inner tag deletion and if there is only one tag in the packet, then the MAC does not delete the tag.
VLAN tag insertion or replacement	MAC_VLAN_Incl[VLC] or MAC_Inner_VLAN_Incl[VLC] respectively enables the VLAN tag insertion or replacement for an outer or inner tag. When you enable the VLAN tag insertion or replacement, the MAC_VLAN_Incl[VLTI] or

Table continues on the next page...

Table 469. Double VLAN processing features in transmit path (continued)

	MAC_Inner_VLAN_Incl[VLTI] determines whether the VLAN tag must be fetched from the register or from the control word.

72.7.1.2 Receive path

Table 470. Double VLAN processing in receive path

Feature	Description
Outer or inner VLAN tag-based filtering	Mac can filter packets through the ERIVLT bit on the basis of the outer or inner VLAN tag.
C-VLAN or S-VLAN tag-based filtering	Mac can filter packets through the ERSVLM bit on the basis of the C-VLAN or S-VLAN type.
Outer and inner VLAN tag stripping	Mac can strip the outer and inner VLAN tags from received frame on the basis of the EVLS and EIVLS bits.
16-bit outer and inner VLAN tag and type in Rx status	Mac can provide the 16-bit outer and inner VLAN tag and type in the Rx status on the basis of the EVLRXS and EIVLRXS bits, respectively.
Disabling or skipping checking of outer VLAN tag type	Mac can disable or skip checking of an outer VLAN tag type to match C-VLAN or S-VLAN on the basis of the DOVLTC bit.

72.7.1.3 Source address and VLAN insertion, replacement, or deletion

This module supports the SA, and VLAN insertion, replacement and deletion feature. The SA and VLAN fields are Tx packet-related control information that interfaces provide as part of the control word. IP supports the feature to insert or replace the source address on the basis of the information in the MAC address registers, and it also supports the feature to insert, replace, or delete the VLAN fields (VLAN type and VLAN tag) on the basis of the setting of the MAC_VLAN_Incl[VLTI]. SA insertion or replacement feature is enabled for all the transmit packets or selective packets. Similarly, VLAN insertion, replacement, or deletion feature is enabled for all the Tx packets or selective packets.

72.7.1.4 Programming source address insertion or replacement

You can use the SA insertion or replacement feature to instruct MAC to perform the following actions for Tx packets:

- Insert the content of the MAC address registers in the SA field
- Replace the content of the SA field with the content of the MAC address registers

When SA insertion is enabled, the application must ensure that the packets sent to MAC do not have the SA field. The MAC does not check whether the SA field is present in the transmit packet and it inserts the content of MAC address registers in the SA field. Similarly, when the SA replacement is enabled, the application must ensure that the SA field is present in the packets sent to the MAC.

MAC replaces the six bytes following the DA field in the transmit packet with the content of the MAC address registers.

- Configure MAC_Configuration[SARC] to enable the SA insertion or replacement for all Tx packets.
- Enable the SA insertion for selective packet by configuring the SA insertion control field (bits [25:23] of TDES3) in the first transmit descriptor of the packet. When you write 1 to bit 25 of TDES3, the SA insertion control field indicates insertion or replacement by MAC Address1 registers. When bit 25 of TDES3 resets, it indicates insertion or replacement by MAC Address 0 registers.

If you do not enable the MAC Address1 registers, then the MAC Address0 registers are used for insertion or replacement irrespective of the value of the most-significant bit of the SA insertion control field.

72.7.1.5 Programming VLAN insertion, replacement, or deletion

You can use the VLAN insertion, replacement, or deletion feature to instruct MAC to perform the following actions for the Tx packets:

- Delete the VLAN type and VLAN tag fields.
- Insert or replace the VLAN type and VLAN tag fields.

Insertion or replacement is done on the basis of the setting of the MAC_VLAN_Incl[VLTI] as described in theTable 471:

Table 471. VLAN insertion or replacement based on VLTI bit

Condition	Description
VLTI bit=1	MAC inserts or replaces the following: • VLAN type field (C-VLAN or S-VLAN as indicated by the MAC_VLAN_Incl[CSVL]) • VLAN tag field depending on the content of the VT field of transmit context descriptor of the packet
VLTI bit resets	MAC inserts or replaces the following: • VLAN type field (C-VLAN or S-VLAN as indicated by the MAC_VLAN_Incl[CSVL]) • VLAN tag field with the MAC_VLAN_Incl[VLT]

When the VLAN replacement or deletion is enabled, MAC checks whether the VLAN type field (0x8100 or 0x88a8) is present after the DA and SA fields in the transmit packet. The replace or delete operation does not occur if the VLAN type field is not detected in two bytes following the DA and SA fields. However, when the VLAN insertion is enabled, MAC does not check the presence of VLAN type field in the transmit packet and just inserts the VLAN type and VLAN tag fields.

Configuring MAC_VLAN_Incl[VLC] and MAC_VLAN_Incl[VLP] enables the VLAN insertion, replacement, or deletion feature for all Tx packets.

Configuring the VTIR field of TDES2 normal descriptor enables the VLAN insertion, replacement, or deletion for selective packets.

In addition, the VLP (VLAN Priority control) field must reset in MAC_VLAN_Incl (for outer VLAN) and MAC_Inner_VLAN_Incl register (in inner VLAN) for MAC to take the control inputs from the host, depending on the configuration.

72.8 Packet Filtering

This section and all its sub-sections are Synopsys proprietary. Used with permission.

MAC receiver contains various packet filtering scheme. Figure 2 shows the filtering sequence in their precedence order of received packet.

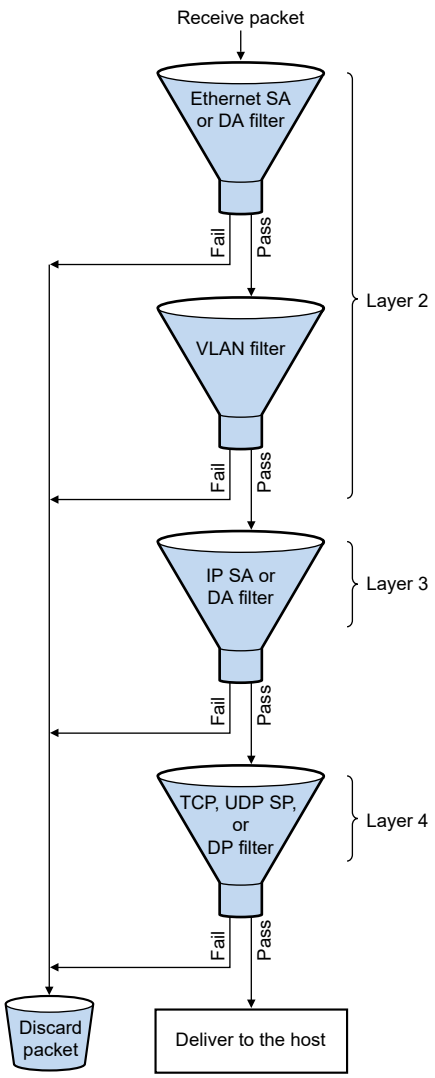


Figure 391. Packet filtering sequence

If you do not enable any of the layer filters, then that filter is bypassed and the subsequent filter is applied. Discard the packet that fails any of the filters. However, you can forward the discarded packet to the host on the basis of the register control.

72.8.1 Source address or destination address filtering

The address filtering module of MAC checks the SA and DA fields of each incoming packet. Programming of different types of address filtering is described in the sections that follows.

72.8.1.1 Unicast destination address filtering

MAC supports up to three MAC addresses for unicast perfect filtering. If perfect filtering is selected (MAC_Packet_Filter[HUC] resets), MAC compares all 48 bits of received unicast address with the programmed MAC address for any match. Default MacAddr0 is always enabled. MAC selects the MacAddr1 to MacAddr2 addresses with an individual enable field. Each byte of MacAddr1 to MacAddr2 can be masked when you compare them with corresponding received DA byte by writing 1 to the corresponding Mask byte control field in the register. This enables group address filtering for the DA.

In hash filtering mode (when you write 1 to HUC bit), MAC performs imperfect filtering by using a 64-bit hash table for unicast addresses . For hash filtering, MAC uses the upper 6 bits CRC of the received destination address to index the content of the hash table. A value of 00000 selects bit 0 of selected register, and a value of 11111 selects bit 63 of hash table register. If the value of

the corresponding bit (indicated by the 6-bit CRC) is 1, the unicast packet is considered to have passed the hash filter, otherwise, the packet is considered to have failed the hash filter.

72.8.1.2 Multicast destination address filtering

Enable MAC_Packet_Filter[PM] to pass all the multicast packets or disable to do multicast addresses filtering on the basis of MAC_Packet_Filter[HMC]. Compare the multicast address with the configured MAC destination address registers (1–2), also supports the group address filtering.

In Hash filtering mode, MAC performs imperfect filtering using a 64-bit hash table. MAC uses the upper 6-bits CRC of received multicast address to index the content of the hash table. A value of 000000 selects bit 0 of selected register and a value of 111111 selects bit 63 of the hash table register. The multicast packet is considered to have passed the hash filter if the value of the corresponding bit is equal to 1. Otherwise, the packet is considered to have failed the hash filter.

72.8.1.3 Hash or perfect address filtering

Configure MAC_Packet_Filter[HPF] to enable or disable the destination address filtering either by hash filter or perfect filter. Also configure MAC_Packet_Filter[HUC] and MAC_Packet_Filter[HMC] to select the hash filtering or perfect filtering for unicast or multicast packet.

72.8.1.4 Broadcast address filtering

MAC does not filter any broadcast packets by default. You must write 1 to MAC_Packet_Filter[DBF] to reject the broadcast packets.

72.8.1.5 Unicast source address filtering

MAC by default compares the source address field with the value configured in the source address registers. Configure bit 30 of MAC address register [1-2] to use it for source address instead of destination address comparison. MAC also supports group filtering with SA. You can filter a group of addresses by masking one or more bytes of the address. MAC drop the packets that fail the SA filter if you write 1 to MAC_Packet_Filter[SAF] field. Otherwise, the result of the SA filter is given as a status bit in the receive status word. When you write 1 to the SAF field, the SA filter and DA filter results ends to decide whether you can forward the packets. This means that the packet is dropped if either filter fails. The packet is forwarded to the application only if the packet passes both filters in-order.

72.8.1.6 Inverse filtering

For DA and SA filtering, you can invert the filter-match result at the final output by writing 1 to the DAIF and SAIF fields of MAC_Packet_Filter register. The DAIF field is applicable for both the unicast and multicast DA packets. The unicast or multicast destination address filter result is inverted in this mode. Similarly, writing 1 to the SAIF field reverses the result of unicast SA filter.

Table 472 and Table 473 summarize the DA and SA filtering on the basis of the type of packets received.

Table 472. Destination address filtering

Packet type	PR	HPF	HUC	DAIF	HMC	PM	DBF	DA filter operation
Broadcast	1	X	X	X	X	X	X	Pass
	0	X	X	X	X	X	0	Pass
	0	X	X	X	X	X	1	Fail
Unicast	1	X	X	X	X	X	X	All packets pass
	0	X	0	0	X	X	X	Pass on perfect or group filter match

Table continues on the next page...

Table 472. Destination address filtering (continued)

	0	X	0	1	X	X	X	Fail on perfect or group filter match
	0	0	1	0	X	X	X	Pass on hash filter match
	0	0	1	1	X	X	X	Fail on hash filter match
	0	1	1	0	X	X	X	Pass on hash or perfect or group filter match
	0	1	1	1	X	X	X	Fail on hash or perfect or group filter match
Multicast	1	X	X	X	X	X	X	Pass all packets
	X	X	X	X	X	1	X	Pass all packets
	0	X	X	0	0	0	X	Pass on perfect or group filter match and drop pause packets if PCF = 0x
	0	0	X	0	1	0	X	Pass on hash filter match and drop pause packets if PCF = 0x
	0	1	X	0	1	0	X	Pass on hash or perfect or group filter match and drop pause packets if PCF = 0x
	0	X	X	1	0	0	X	Fail on perfect or group filter match and drop pause packets if PCF = 0x
	0	0	X	1	1	0	X	Fail on hash filter match and drop pause packets if PCF = 0x
	0	1	X	1	1	0	X	Fail on hash or perfect/group filter match and drop pause packets if PCF = 0x

Table 473. Source address filtering

Packet type	PR	SAIF	SAF	SA filter operation
Unicast	1	X	X	Pass all packets.
	0	0	0	Pass status on perfect or group filter match but do not drop packets that fail
	0	1	0	Fail status on perfect or group filter match but do not drop packet
	0	0	1	Pass status on perfect or group filter match and drop packets that fail
	0	1	1	Fail status on perfect or group filter match and drop packets that fail

NOTE

When you write 1 to the MAC_Packet_Filter[RA] field, all packets are forwarded to the system along with the correct result of the address filtering in the Rx Status.

72.8.2 VLAN filtering

VLAN tag filter can be done either by perfect match or hash table. Depending upon the configuration, MAC can compare either lower 12 bits or all 16 bits of received VLAN tag for perfect match. MAC forward the VLAN tagged packet with match status when it drop all the packets which do not match.

MAC uses 16-bit VLAN hash table for group address filtering on the basis of the VLAN tag. If MAC enables the hash filtering then the most significant 4-bit of CRC-32 of VLAN tag are used to index the content of the MAC_VLAN_Hash_Table. A value of 1 in the VLAN hash table register, corresponding to the index indicates that the VLAN tag of the packet has matched and the packet should be forwarded. A value of 0 indicates that VLAN-tagged packet should be dropped.

MAC supports the inverse matching for VLAN packets. In the Inverse matching mode, the packet must be dropped when the VLAN tag of a packet matches the perfect or hash filter. If MAC enables the VLAN perfect and VLAN hash match, then a packet is considered as matched if either the VLAN hash or the VLAN perfect filter matches. When inverse match is set, MAC forwards a packet only when both the perfect and hash filters indicate mismatch.

Table 474 specifies the different possibilities for VLAN matching and the final VLAN match status. When you write 1 to MAC_Packet_Filter[RA] all packets are received and it indicates the VLAN match status in the VF field of RDES2 normal descriptor (write-back format). When the RA field is not set and when you write 1 to MAC_Packet_Filter[VTFE] the packet drops if the final VLAN match status fails

When VLAN VID programs to 0 in the VL field of MAC_VLAN_Tag register, consider that all the VLAN-tagged packets are perfectly matched but the status of the VLAN hash match depends on the VTHM and VTIM fields in MAC_VLAN_Tag register.

Table 474. VLAN match status

VID	VLAN perfect filter match result	VTHM bit	VLAN hash filter match result	VTIM bit	Final VLAN match status
VID = 0	Pass	0	X	X	Pass
	Pass	1	X	0	Pass
	Pass	1	Fail	1	Pass
	Pass	1	Pass	1	Fail
VID1= 0	Pass	X	X	0	Pass
	Fail	0	X	0	Fail
	Fail	1	Fail	0	Fail
	Fail	1	Pass	0	Pass
	Fail	0	X	1	Pass
	Pass	X	X	1	Fail
	Fail	1	Pass	1	Fail
	Fail	1	Fail	1	Pass

In Table 474, X represents any value.

72.8.3 VLAN filter fail packets queue

When you enable VLAN filtering, route the VLAN filter fail packets to a programmable queue (VFFQ) when the value of MAC_Packet_Filter[RA] = 1 or MAC_Packet_Filter[VTFE] = 0 and write 1 to MAC_RxQ_Ctrl4[VFFQE] for the queue.

Route the packets that passes the VLAN filtering to Rx queues on the basis of the VLAN tag priority field by configuring the PSRQ field in the corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. Discard the packets that fail the VLAN filter if RA=0 or VTFE=1. However, when RA=1 or VTFE=0, forward the VLAN filter fail packets to the application. In such case, when you write 1 to the VLAN filter fail queue enable (VFFQE) field, the VLAN filter fail packets forward to the Rx queue number programmed in the VFFQ field. If the value of VFFQE=0, the VLAN priority mapping determines the Rx queue number per the PSRQ fields.

Table 475 shows the Rx queue routing table for unicast tagged packets, with DA or SA filter enabled.

Table 475. Rx queue routing table for unicast tagged packets

RA	VFTE	SA or DA filter result	VLAN filter result	VFFQE	Queue routing
X	X	Pass	Pass	X	PSRQ
0	0	Pass	Fail	0	PSRQ
0	0	Pass	Fail	1	VFFQ
0	X	Fail	X	X	Dropped
0	1	Pass	Fail	X	Dropped
1	X	Fail	X	0	UFFQ*/PSRQ
1	X	Fail	X	1	UFFQ*/VFFQ
1	X	Pass	Fail	0	PSRQ
1	X	Pass	Fail	1	VFFQ

X : Don't care condition * : When UFFQE is enabled else PSRQ.

72.8.4 Extended receive VLAN filtering and routing

When the extended Rx VLAN filtering and routing is enabled then both the perfect filtering and hash filtering can be enabled. The overall VLAN filter result is based on the perfect filter result and hash filter result (if enabled). Filter result is passed to application as part of the status bit. Extended routing will take place only if VLAN filter has passed. Routing is only based on perfect filter result. Each perfect filter has a DMA channel enable and DMA channel number filed which must be programmed for routing.

See the extended VLAN based DMA selection in Dynamic (per packet) mapping for more information about routing.

72.8.4.1 Comparison mode

Application has these comparison option for each VLAN tag filter:

- Programs MAC to compare the inner or outer VLAN tag either with 12 bits or 16 bits programmed VID.
- Selects whether the VID comparison is for SVLAN or CVLAN type frames, if type check is enabled for a filter.

72.8.4.2 Filtering

Perfect filtering is done on the basis of the MAC_VLAN_Tag_Filter registers. MAC compares the relevant VLAN tag ID and gives a result for each VLAN tag filter.

The results for each VLAN tag filter are:

- Pass- If any one of the VLAN tag filters gives a match.
- Fail- If the frame mismatches all the filters.

This behavior is applicable only when the inverse filtering is not enabled in MAC_VLAN_Tag_Ctrl.

If inverse filtering is enabled and the frame mis-matches all the relevant filters then it is considered to have passed the VLAN filter. If the frame matches any one of the relevant filters then it is considered to have failed. The frame is bypassed to the application if none of the enabled filters can perform a comparison or if none of the filters are enabled.

The overall filter result and the programming on MAC_Packet_Filter[VTFE] and MAC_Packet_Filter[RA] determines if the frame will drop or it is forwarded to the application. If the value of RA = 1 or VTFE = 0, then the frame is always forwarded whether the filter result is a pass or fail, . If the value of RA = 0 and VTFE = 1, only then, if the VLAN tag filter result is a pass the MAC forwards the frame. If the frame is forwarded to the application, then the relevant filter result is indicated through the status bits.

72.8.4.3 Filter status

The extended receive VLAN filtering and routing feature provides two status fields to show the comparison result of the VLAN tags.

By default, MAC indicates the VLAN filter status through one bit in the status VF field in RDES2. When you enable the extended RX VLAN filtering and routing, two status fields will show the comparison result of the VLAN tags. The outer VLAN tag filter pass and inner VLAN tag filter pass bits are defined in the following positions. The status indicated through these fields depends on the programming as described below.

In RDES2:

- Bit 15 – Outer VLAN tag filter status
- Bit 14 – Inner VLAN tag filter status

In ARI status: MAC filter status:

- Bit 15 – Outer VLAN tag filter status
- Bit 14 – Inner VLAN tag filter status

In MRI status:

- Bit 47 – Outer VLAN tag filter status
- Bit 46 – Inner VLAN tag filter status

Outer VLAN tag filter status (OTS)

- In perfect filtering, without enabling inverse filtering, if you write 1 to this field, it indicates that the frame's outer VLAN tag has matched one of the VLAN tag filters.
- If this field resets, it indicates that the frame's outer VLAN tag has either failed the relevant outer VLAN tag filters or bypassed them.
- This field resets if none of the filters are enabled for outer VLAN tag comparison.
- If inverse filtering is enabled and if you write 1 to this field, then the frame's VLAN tag has passed all the relevant VLAN tag filters. If it is resets, then it has failed at least one filter or bypassed all the filters programmed for outer VLAN tag comparison.
- This bit is valid for both single and double VLAN tagged frames.

Inner VLAN tag filter status (ITS)

- In perfect Filtering, without enabling inverse filtering, if you write 1 to this field, it indicates that the frame's inner VLAN tag has matched one of the VLAN tag filters.
- If this field resets, it indicates that the frame's inner VLAN tag has either failed the relevant inner VLAN tag filters or bypassed them.
- This field resets if none of the filters are enabled for inner VLAN tag comparison.
- If Inverse Filtering is enabled and if you write 1 to this field, then the frame's VLAN tag has passed all the relevant VLAN tag filters. If it resets, then it has failed at least one filter or bypassed all the filters programmed for inner VLAN tag comparison.

- This bit is valid for only double VLAN tagged frames, when double VLAN processing is enabled.

The application must observe the status fields and the program to determine if the frame has passed or failed the VLAN filter.

Table 476 and Table 477 describes the possible filter combinations and the corresponding filter results and they also explain the scenarios when the double VLAN processing and the hash VLAN filter are enabled in the design.

Legend for the table OTS and ITS Bit Values with at least 1 perfect filter enabled

- VTIM: VLAN tag inverse match enable – bit 17 in VLAN_Tag_Ctrl register.
- HFO: Hash filter enabled for outer VLAN tag comparison - bit 25 and bit 27 in VLAN_Tag_Ctrl register.
- HFI: Hash filter enabled for inner VLAN tag comparison – bit 25 and bit 27 in VLAN_Tag_Ctrl register.
- PFO – Perfect filter comparison enabled for outer VLAN tag - Any of the MAC_VLAN_Tag_Filter registers is enabled (write 1 to bit 16) and programmed for outer VLAN tag comparison (write 0 to bit 20).
- PFI – Perfect filter comparison enabled for Inner VLAN Tag - Any of the MAC_VLAN_Tag_Filter registers is enabled (write 1 to bit 16) and programmed for inner VLAN tag comparison (write 1 to bit 20).
- OTS – Outer VLAN tag filter status
- ITS – Inner VLAN tag filter status

Table 476 shows the possible values of status bits (OTS and ITS) when at least one perfect filter is enabled.

Table 476. OTS and ITS bit values with at least 1 perfect filter enabled

VTIM	HFO	HFI	PFO	PFI	OTS	ITS
0	0	0	0	1	0	1/0
0	0	0	1	0	1/0	0
0	0	0	1	1	1/0	1/0
0	1	0	1	1	1/0	1/0
0	1	0	1	0	1/0	0
0	1	0	0	1	1/0	1/0
0	0	1	1	1	1/0	1/0
0	0	1	1	0	1/0	1/0
0	0	1	0	1	0	1/0
1	0	0	0	1	0	1/0
1	0	0	1	0	1/0	0
1	0	0	1	1	1/0	1/0
1	1	0	1	1	1/0	1/0
1	1	0	1	0	1/0	0
1	1	0	0	1	1/0	1/0

Table continues on the next page...

Table 476. OTS and ITS bit values with at least 1 perfect filter enabled (continued)

VTIM	HFO	HFI	PFO	PFI	OTS	ITS
1	0	1	1	1	1/0	1/0
1	0	1	1	0	1/0	1/0
1	0	1	0	1	0	1/0

Table 477 shows the possible values of status bits (OTS and ITS) when no perfect filters are enabled except the VLAN hash filter is enabled.

Table 477. OTS and ITS bit values with only VLAN hash filter enabled

VTIM	HFO	HFI	OTS	ITS
0	0	0	0	0
0	1	0	1/0	0
0	0	1	1/0	1/0
1	0	0	1/0	0
1	1	0	1/0	0
1	0	1	1/0	1/0

With no perfect filters enabled, any VLAN packet is considered to have bypassed the perfect filter. If Hash Filter is enabled for one of the Tags, then the respective Status bit depends on the Filter's result. The Status bits are set to 0 if VLAN Hash Filter is also not enabled.

If the value of ITS/OTS is shown as 1/0; then it indicates that the final result is dependent on the enabled relevant filter's result.

Example 1: The second row of OTS and ITS bit values with only VLAN hash filter enabled indicates that at least one Perfect Filter is enabled for Outer VLAN tag comparison and none of the filters are enabled for Inner VLAN tag comparison. Inverse VLAN Filtering is not enabled. The bit OTS is given as 1/0. If the received frame passes atleast one of the enabled Outer VLAN Tag filters then the bit is set to 1. If the frame doesn't pass any of the enabled Outer VLAN Tag filters, then the bit is set to 0.

Example 2: The last row of table "OTS and ITS bit values with only VLAN hash filter enabled" indicates that the inverse filtering is enabled, hash filter and at least one perfect filter is enabled for Inner VLAN Tag comparison, then if the received frame's Inner VLAN tag mismatches with both the Hash Filter and all the enabled Perfect filters, then the frame will have the ITS bit set to 1 else it is set to 0. OTS will be set to 0 as no comparison is performed.

72.8.4.4 Stripping

Each VLAN tags has individual control over stripping. The programming options of always strip, never strip, strip on pass and strip on fail are available. Inner or outer VLAN tag stripping is based on the pass or fail results of the individual tag. If all the relevant filters bypasses a tag, stripping is not applicable for the tag.

- If strip on pass is enabled for the outer VLAN tag, then the stripping occurs only if the outer VLAN tag has passed the relevant filters. The outer VLAN tag filter result field will be 1.
- If strip on fail is enabled for the outer VLAN tag, then the stripping occurs only if the outer VLAN tag has failed relevant filters. The outer VLAN tag filter result field will reset.
- If the outer VLAN tag of the received frame is bypassed by the entire filter (no comparison has been made), then the tag is not stripped, though the status bit is 0.

- As multiple filters are enabled, it is possible that the received VLAN frame could have matched any one or more of thefilters. The VLAN Tag's value is not always deterministic from the filter status bits.

If an application strips the VLAN tag on the basis of the filter result, it might lose the VID. You can use it, if stripping is enabled for any of the tags, place the tag in the status. For this the software or application will enable the respective VLAN tag in status bit - 24 or 31 in the MAC VLAN tag control register.

See section Programming guidelines for extended VLAN filtering and routing on receive for more information.

72.8.5 Layer 3 and layer 4 filtering

IP supports the layer 3 based packet filtering which is an IP source or destination address filtering in the IPv4 or IPv6 packets, and layer 4 based packet filtering which is a source or destination port number filtering in TCP or UDP. When you enable layer 3 and layer 4 packet filtering then packets are filtered in the following manner.

- Matched packets: MAC forward the packets that match all enabled fields to the application along with the status. It gives the matched field status only if MAC_Configuration[IPC] = 1 and if one of the following conditions is true:
  - All enabled layer 3 and layer 4 fields match
  - At least one enabled field matches and other fields are bypassed or disabled

When you enable multiple layer 3 and layer 4 filters, any filter match is considered as a match. If more than one filter matches, MAC provides the status of the lowest filter where filter 0 is the lowest filter and filter 3 is the highest filter.

- Unmatched packets: MAC drop the packets that do not match any enabled fields. You can use the inverse match feature to block or drop a packet with specific TCP or UDP over IP fields and forward all other packets.
- Non-TCP or UDP IP packets: By default, all the non-TCP or UDP IP packets bypasses the layer 3 and layer 4 filters. You can optionally program MAC to drop all the non-TCP or UDP packets over IP packets.
- IP has a control register MAC_L3_L4_Control to control the layer L3 and L4 packet filtering along with address registers to program the layer L3 and L4 fields to be matched.

72.8.6 Flexible receive parser

When you enable this function, all incoming packets are parsed per the programmable instruction stored in memory. It perform these functions:

- Packet filter (Accept or reject)
- DMA channel selection

Flexible receive parser block operates over the first 128 bytes data of receive packet on the basis of 64 bytes of instruction on each packet. Figure 392 shows the functional block diagram of flexible receive parser.

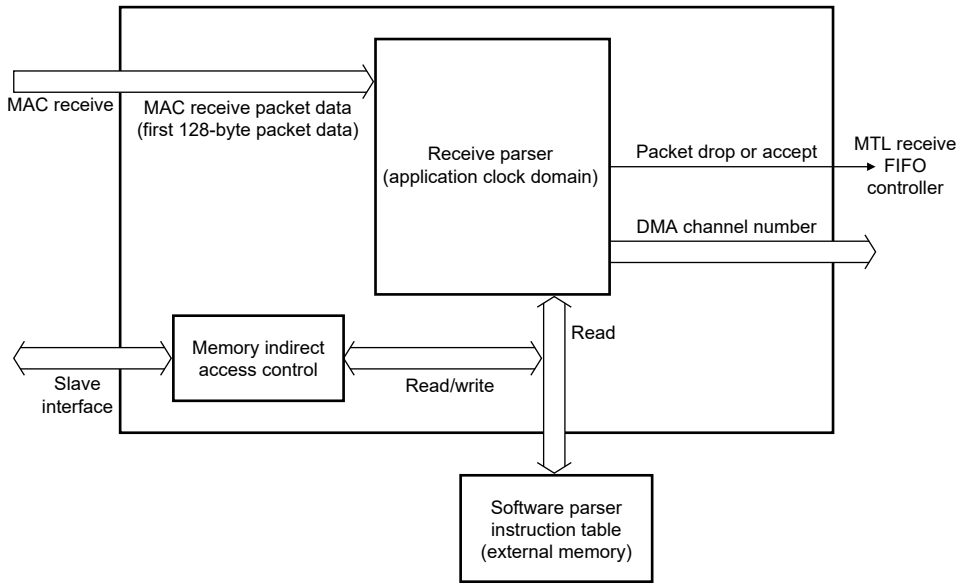


Figure 392. Functional block diagram of flexible recieve parser

72.8.6.1 Instruction table format of the flexible receive parser

You must build the 96-bit wide instruction before enabling the receive parser feature and program each instruction as an entry in the instruction memory.

Table 478 shows the format of each entry in the instruction table .

Table 478. Rx parser instruction entry format

Field	Field Name	Description
31:0	MATCH_DATA	This is a 4-byte data. You can use it to compare with incoming packet data which starts at the frame offset as defined in [79:72] of the entry. The comparison is done only on those bits whose corresponding mask bits are 1 (MATCH_EN bits of [63:32]). See the notes (below this table) on the byte order relative to Ethernet arrival data and endianness which the QoS IP supports on the slave interface. The programming of this instruction table must adhere to the same.
62:32	MATCH_EN	When the MATCH_EN = 1, you can use the corresponding packet data bit for comparing. Otherwise, corresponding data bit is don’t care.
64	AF	Accept frame
65	RF	Reject frame
66	IM	Inverse match
67	NC	Next instruction control
71:68	Reserved	-

Table continues on the next page...

Table 478. Rx parser instruction entry format (continued)

Field	Field Name	Description
79:72	Frame offset	[77:72] indicates the frame offset in terms of 4 bytes. (Here [79:78] always 0) Conpare the frame offset in the packet data for Match.This is in terms of 4 bytes. The value of actual frame offset in bytes are 0 01 42 8… ..63 252 The max value programmed in this instruction table is 128 B. The 'Actual frame offset' must not cross this limit.
87:80	OK index	If (NIC==0) the memory index you can use next when ENTRY_MATCH==1 and neither AF or RF = 1. If (NIC==1) the memory index you can use next when ENTRY_MATCH==0
95:88	DMA Ch No	Indicates the DMA channel number (1-bit for each). You can use this when ENTRY_MATCH ==1 and AF = 1 (Accept frame). The following bits give the encoding: bit[0]- DMA channel number o, bit[1]- DMA channel number , 1bit[2]- DMA channel number 2… bit[7]- DMA channel number 7. **NOTE** You can encode the DMA channel number using bit wise as QoS IP support multicasting or broadcast across DMA channels. See section 3.3.12 for more information and proper usage.
128:96	Reserved	0 (This field is only for software view and reserved for future enhancements. The memory width remains as 96-bits).

The parser begins parsing from the 0th entry, for each received packet. The subsequent parsing entry location (next entry or OK_INDEX[]) depends on the current entry parser result. Flexible receive parser flow chart shows the detailed flow.

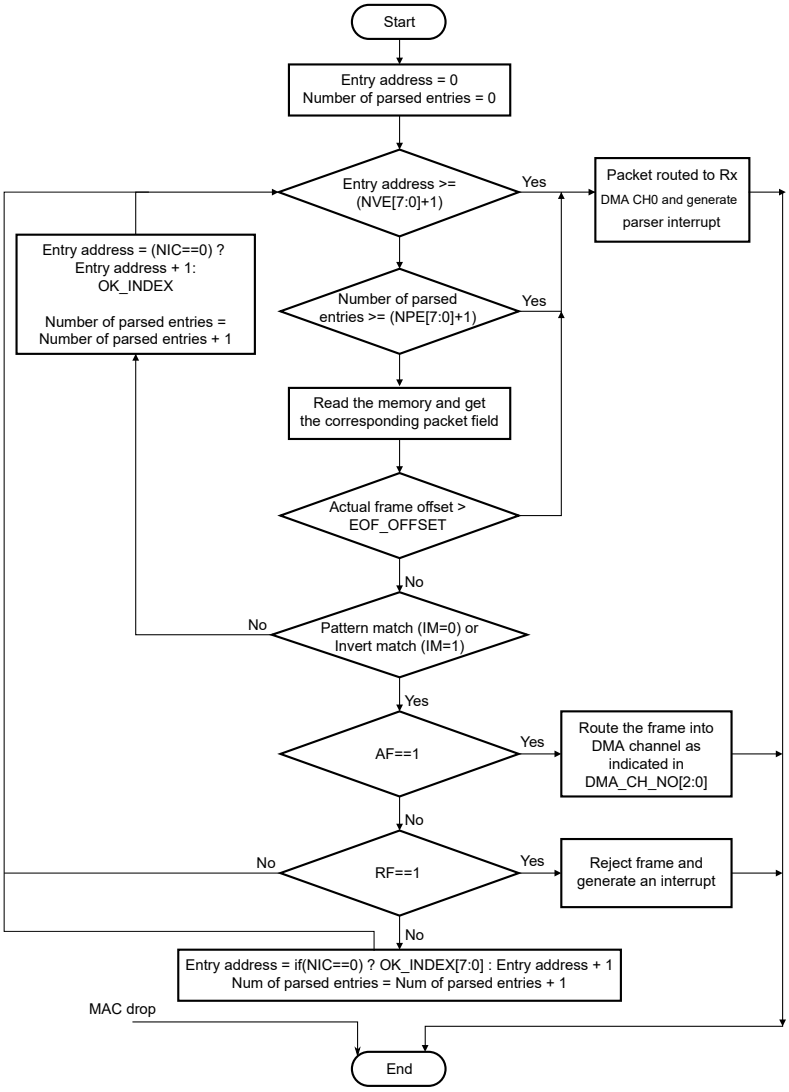


Figure 393. Flexible recieve parser flow chart

The EOF_OFFSET shown in the figure is the least among the following:

- Actual packet size
- 128 Byte when frame pre-emption feature is not enabled or configured
- 128 Byte when frame pre-emption feature is enabled and packet is pre-emptible
- 64 Byte when frame pre-emption feature is enabled and packet is express packet

72.8.6.2 Number of valid entries (NVE)

NVE indicates the number of valid entries in the instruction table. The default value is 64. You can change this value by writing into MTL_RXP_Control_Status[NVE].

When parsing, an error is flagged in MTL_RXP_Interrupt_Control_Status[NVEOVIS], if the entry address is greater than MTL_RXP_Control_Status[NVE] + 1. IP generates an interrupt status (MTL_Interrupt_Status[MTLPIS]).

72.8.6.3 Number of parsable entries (NPE)

NPE indicates the number of entries that you can parse on an incoming packet. The default value is 64. Write into MTL_RXP_Control_Status[NPE] to change this value.

When parsing, an error is flagged in MTL_RXP_Interrupt_Control_Status[NPEOVIS], if the number of parsed of parsed entries is more than MTL_RXP_Control_Status[NPE] + 1. IP generates an interrupt status (MTL_Interrupt_Status[MTLPIS]).

NVE[] indicates the number of valid entries in the receive parser memory which you have built and NPE[] indicates the worstcase parsable entries considering the various possible paths that the parser can take. (NVE[] >= NPE[]).

72.8.6.4 Frame offset

It indicates the frame offset, in terms of 4 bytes which you use when comparing against the current table entry.

In an entry, an error is flagged in MTL_RXP_Interrupt_Control_Status[FOOVIS] in any of the following cases:

When frame offset is more than:

- Packet size
- 128B if pre-emption is disabled
- 128B if pre-emption is enabled and packet is pre-empitble
- 64B if pre-emption is enabled and packet is an express packet

IP generates an interrupt status (<xref href="DWC_ether_qos.dita#MTL_Interrupt_Status/MTL_Interrupt_Status.MTLPIS" >MTL_Interrupt_Status[MTLPIS]</xref>).

NOTE

If the frame ends after the (frame_offset *4) bytes but before ((frame_offset *4)+4) bytes, the receive parser declares a frame offset error if you enable comparison (via MASK bits) for the non-received bytes; from (frame_offset *4) to ((frame_offset *4)+4).

72.8.6.5 Frame reject

You can drop the frame in cases, where an entry matches (considering the inverse match, IM bit in the instruction table) and frame accept bit is not 1, and frame reject bit (RF) = 1.

When a frame drops because RF = 1, IP generates an interrupt status (MTL_Interrupt_Status[MTLPIS]).

72.8.6.6 Out of order processing

The Rx parser flow can be out of order. The decision tree is not based on the order in which packet arrives. This implies that the next entry and next frame offset (OK_INDEX[] and FRAME_OFFSET[]) can be less than the current entry address and current frame offset.

72.8.6.7 DMA channel selection

When you enables the receive parser (MTL_Operation_Mode[FRPE] = 1) it overrides the MAC DMA selection criteria.

When you disables the receive parser (MTL_Operation_Mode[FRPE] = 0) the MAC/MTL determines the DMA channel number.

72.8.6.8 Receive queue selection

The existing MAC functionality determines the receive queue, irrespective of whether the receive parser is enabled or disabled. This is because the receive parser is mainly designed to select the DMA channel and the MAC decides the receive queuing to enable the proper functioning of the pause or PFC feature.

NOTE

When there are multiple queues and multiple DMA Channels are enabled, a particular DMA channel might receive out of order packets because queue selection is based on MAC criteria (VLAN Priority and other criteria) and Rx DMA channel selection is based on receive parser.

72.8.6.9 MAC packet filtering/drop/error handling

Packet filtering is done on the basis of the received packet fields. You must disable the packet filtering features inside the MAC when the receive parser is enabled. Follow these steps to disable the packet filtering features inside the MAC :

1. Write 1 to the Promiscuous mode (MAC_Packet_Filter[PR]).
2. Write 1 to all other fields in MAC_Packet_Filter to its default values.

When MAC decides to drop the packet due to the receive MAC dependent error such as:

- GMII error
- Receive watchdog error
- CRC error
- Giant frame error

the packet drops irrespective of the receive parser decision.

72.8.6.10 Pad strip or CRC strip handling

Software controls the pad strip and CRC strip and it is applicable to all the received packets. Also,software must build the receive parser instructions accordingly, when you enable these features.

NOTE

Enable pad and CRC stripping in real use case.

72.8.6.11 VLAN strip handling

MAC supports stripping the outer VLAN as well as inner VLAN. When you program MAC to strip the VLAN, the receive parser considers those VLAN tags to be part of the incoming packet.

72.8.6.12 Multicast and broadcast support

IP supports multicast and broadcast packet and the packet route to the highest queue number.

When you enables the receive parser, the DMA channel numbers (DMA CH NO, Bits [95:88]) in the instruction table, decide the packet routing.

When you disables the receive parser, DCS (DMA Channel Select) field of the MAC_AddressX_High register determines the packet routing. See the Broadcast/multicast packet duplication.

NOTE

You can select multiple DMA channels for the packet routing because the DCS/DMA CH NO field is per the DMA channel control.

72.8.6.13 Pre-emption support

IP supports the frame pre-emption feature when the flexible receive parser feature is enabled. It has the following limitations:

- Number of bytes would be 64, so that the receive parser can operate on an express traffic
- Number of bytes would be 128, so that receive parser can operate on a pre-emptible traffic.

72.8.6.14 Software access to the flexible receive parser memory

Software can read and write into the receive parser memory. It uses the following registers via indirect addressing.

- MTL_RXP_Indirect_Acc_Control_Status
- MTL_RXP_Indirect_Acc_Data

For more details, see EMAC register descriptions.

72.8.6.15 Statistical counters

IP supports these statistics counters for flexible receive parser.

1. MTL_RXP_Drop_Cnt
2. DMA_RXP_Error_Cnt

DMA_CH(#i)_Rxp_Accept_cnt (i=0; i<2)

For more details, see EMAC register descriptions.

72.8.6.16 Changing the instruction table by software

Software can update the instruction table in the memory in the following ways:

- Disables the MAC receiver and receive parser by writing 0 to MAC_Configuration[RE] and MTL_Operation_Mode[FRPE]. It waits for receive parser to become inactive (RXPIA, MTL_RXP_Control_Status register bit[31]).
- (Optional) Programs in the entry address 0 to unconditionally (all MATCH_EN bit 0) skip to OK_INDEX[] (to certain location in the memory), where you can program it to reject or accept all the packets.

72.8.6.17 Receive cut-through functionality

In the Cut-Through mode, the receive controller transfer the received packets to DMA just after it receives the cut-through threshold amount of packet data (RTC field in the MTL_RxQx_Operation_Mode register).

However due to Rx parser's result worst case arrival time can be more than the programmed cut-through threshold. So, the receive controller delays the packet transfer to DMA accordingly, and the cut-through functionality is re-defined as follows:

The receive controller transfers packet to DMA after at least RTC number of bytes have been received and receive parser results are available.

72.8.6.18 Receive packet drop indication

In most cases, the packet drops internally in the receive queue. However, back-to-back packets can arrive for the same queue and the first packet's receive parser result is not available when the second packet arrives. In such cases, the packet cannot be flushed internally and forwarded to DMA or the application interface and indicates the status accordingly. See RDES2, bit 16(RXPD) in Receive normal descriptor (write-back format) for more information.

72.8.6.19 Runt packet handling

The receive parser can performs the operation with 64B minimum size packet. If the runt packet is received (MAC must indicate it), the receive parser assumes that it is dropped in the MAC, this is because, the back-to-back runt packets (< 64B) cannot be handled in the receive parser.

72.8.6.20 Uncorrectable ECC error handling

The packet does not drops when the parser detects an uncorrectable ECC error while parsing the parser memory. The packet is sent to the application with an error indication. The RXPI (RX parser incomplete) sets in the packet status along with error summary (ES) bit.

See the Receive normal descriptor (write-back format) for more information.

72.9 IEEE 1588 timestamp support

This section and all its sub-sections are Synopsys proprietary. Used with permission.

IP supports the IEEE 1588 precision time protocol (PTP). It is also able to do precise time stamping and captures incoming and outgoing frame because of the PTP protocol implementation in the IP. It supports all the clock types defined in IEEE 1588-2008. The typical frequency of PTP timestamp clock is 125 MHz.

IP supports these features:

- Provides an option to take snapshot of all packets or only PTP type packets
- Provides an option to take snapshot of only event messages
- Identifies the PTP message type, version, and PTP payload in packets sent directly over Ethernet and sends the status
- Provides an option to measure sub-second time in digital or binary format

72.9.1 Delay request-response mechanism

The system or network is classified into the master and slave nodes for distributing the timing and clock information. Figure 394 shows the process that PTP uses for synchronizing a slave node to a master node by exchanging PTP messages.

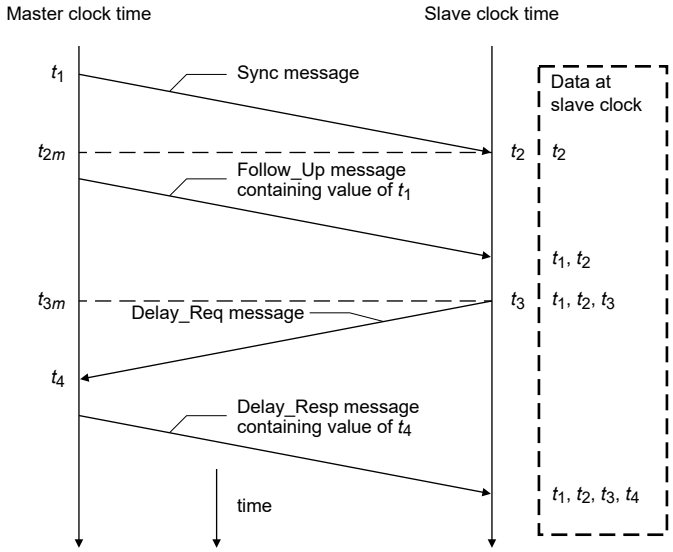


Figure 394. Networked time synchronization

As shown in the figure, PTP uses this process:

1. The master broadcast the PTP sync messages to all its nodes. The sync message contains the reference time information of the master. This message leaves the system of the master at t1. You must capture this time for Ethernet ports at GMII or MII.
2. The slave receives the sync message and it also captures the exact time, t2, using its timing reference.
3. The master sends a Follow_Up message to the slave, which contains t1 information for later use.
4. The slave sends a Delay_Req message to the master and note the exact time, t3, at which this packet leaves the GMII or MII interface.
5. The master receives the message, capturing the exact time t4, at which the message enters its system.
6. The master sends the t4 information to the slave in the Delay_Resp message.
7. The slave use the four values of t1, t2, t3, and t4 to synchronize its local timing reference to the master's timing reference.

Most of the PTP implementation is done in the software above the Ethernet layer. However, the hardware captures the exact time when specific PTP packets enter into or leave the Ethernet port at the MII interface. This timing information must be captured and returned to the software for proper implementation of PTP with high accuracy.

72.9.2 Peer-to-peer PTP transparent clock (P2P TC) message support

The IEEE 1588-2008 supports the peer-to-peer PTP (Pdelay) message in addition to the sync, delay request, follow-up, and delay response messages. Figure 395 shows the method to calculate the propagation delay in clocks supporting the peer-to-peer path correction.

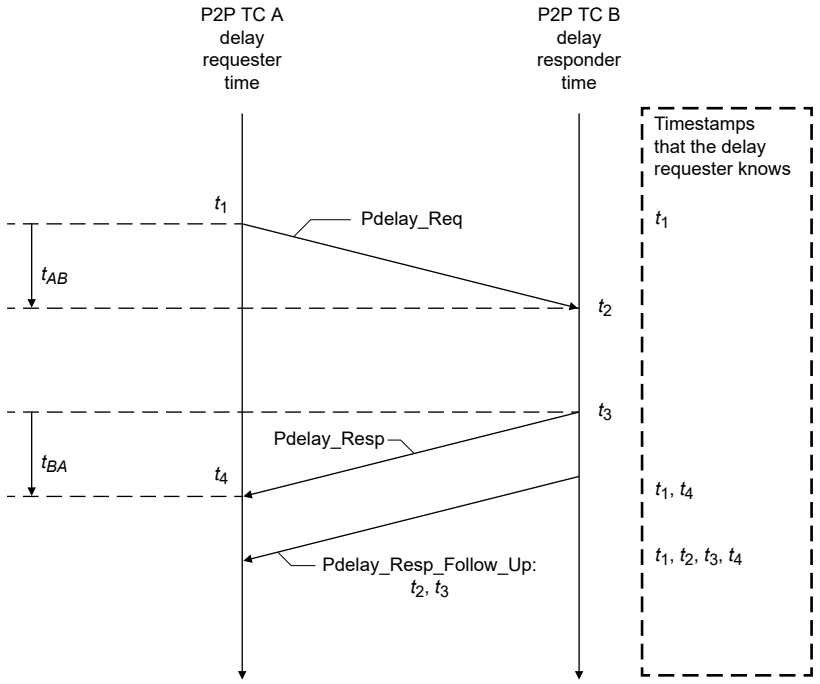


Figure 395. Propagation delay calculation in clocks supporting peer-to-peer path correction

As shown in the above figure, the propagation delay is calculated in the following way:

1. Port 1 issues a Pdelay_Req message and generates a timestamp (t1) for the Pdelay_Req message.
2. Port 2 receives a Pdelay_Req message and generates a timestamp (t2) for this message.
3. Port 2 returns a Pdelay_Resp message and generates a timestamp (t3) for this message. Port 2 returns the Pdelay_Resp message as quickly as possible after the receipt of the Pdelay_Req message, to minimize the errors because of any frequency offset between the two ports. Port 2 returns any one of the following:
   - Difference between the timestamps t2 and t3 in the Pdelay_Resp message
   - Difference between the timestamps t2 and t3 in the Pdelay_Resp_Follow_Up message
   - Timestamps t2 and t3 in the Pdelay_Resp and Pdelay_Resp_Follow_Up messages, respectively
4. Port 1 generates a timestamp (t4) after receiving the Pdelay_Resp message.
5. Port 1 uses all the four timestamps t1, t2, t3, and t4 to compute the mean link delay.

72.9.3 Timestamp correction

According to the IEEE 1588 specification, you must capture a timestamp when the PTP message timestamp point (leading edge of the first bit of the octet immediately following the start frame delimiter octet) crosses the boundary between the node and the network. As MAC takes the timestamp at an internal point far from the actual boundary of the node and network, you must correct or update the captured timestamp for the ingress or egress path latency (including the delay in the PHY layers). Further

correction is done for the inaccuracies or errors introduced because the clock (MII Tx, Rx clock) is different at the capture point as compared to the PTP clock (clk_ptp_ref_i) that generates the time. The resultant CDC (Clock domain crossing) circuits add an error depending on the clock period of the MII and PTP clocks.

72.9.3.1 Ingress correction

The timestamp captured at the internal snapshot point In the receive side is delayed (later in time) as compared to the time at which the packet's SFD bit is received at the port's boundary. Therefore, the ingress latency and the errors in CDC sampling reduces the captured timestamp. You must determine or calculate the correction value and write into the MAC_Timestamp_Ingress_Corr_* registers.

The correction value consists of these three components:

1. External latency in the PHY layer between boundary point and the input of the core.

   If the PHY is compliant with the IEEE 802.3 Clause 45 MMD registers, it has a register which indicates the maximum and minimum ingress latency. You can read these registers and determines the average ingress latency in the PHY. Alternatively (if the PHY does not support these registers), you must determine the ingress latency from its datasheet or timing characteristics.

2. Internal latency from the core's input to the internal capture point.

   You can read the internal ingress latency from MAC_Timestamp_Ingress_Latency. This is a read-only register and provides the latency in scaled nanoseconds format as defined in IEEE 1588 Clause 5.3.2. The latency differs on the basis of the active PHY interface (MII, RMII, so on) and the operating speed. Therefore, you must read this register after any speed change in MAC, to determine the current internal latency.

3. CDC synchronization

   The CDC synchronization error is almost equal to twice the clock-period of the PTP clock (clk_pt-p_ref_i).

You must add the values that these three components determine and must write into the TSIC and TSICSNS fields of the MAC_Timestamp_Ingress_Corr_* registers.

NOTE

The value written into the register must be negative (two's complement as explained below), because it is subtracted from the captured timestamp. The MAC receiver adds the value in this register to the captured timestamp and then gives the resultant value as the timestamp of the received packet.

When MAC_Timestamp_Control[TSCTRLSSR] is 1, it indicates that the nanoseconds field of the captured timestamp is in decimal format with a granularity of 1ns. So the Bit31 of TSIC must be 1 (for negative value) and bits[30:0] must write with "10^9 - total ingress_correction_value[nanosecond part]" represented in binary. For example, if the required correction value is -5 ns, then the value is 0xBB9A_C9FB.

When MAC_Timestamp_Control[TSCTRLSSR] becomes 0, it indicates that the nanoseconds field of the captured timestamp is in binary format with a granularity of ~0.466ns. Therefore, bits[30:0] must write with "2^31 - total ingress_correction_value" represented in binary with bit[31] = 1.

72.9.3.2 Egress correction

The timestamp captured at the internal snapshot point in the transmit side is earlier (advanced in time) as compared to the time at which the packet's SFD bit is output at the port's boundary. Therefore, the egress latency and the errors in CDC sampling must compensate the captured timestamp. You must determine or calculate this correction value and write into the MAC_Timestamp_Egress_Corr_* registers.

The correction value consists of these three components:

1. External latency in the PHY layer between the core output and the port and the network boundary

   If the PHY is compliant with the IEEE 802.3 Clause 45 MMD registers, it has a register which indicates the maximum and minimum egress latency. You can read these registers and determine the average egress latency in the PHY. Alternatively (if the PHY does not support these registers), you must dertermine the egress latency from its datasheet or timing characteristics.

2. Internal latency from the internal capture point and the core output of the c

   You can read this internal egress latency from MAC_Timestamp_Egress_Latency. This is a read-only register and gives the latency in scaled nanoseconds format as defined in IEEE 1588 Clause 5.3.2. The latency differs on the basis of the active PHY interface (MII, RMII, so on) and the operating speed. Therefore, you must read this register after any speed change in MAC to determine the current internal latency.

3. CDC synchronization error

   The CDC synchronization error value for one-step timestamping = (1 * period of clk_ptp_ref_i + 4 * period of clk_tx_i). Otherwise (Two-step timestamping mode), the value = -(2 * period of clk_ptp_ref_i).

72.9.3.3 Frequency range of reference timing clock

The timestamp information is transferred across asynchronous clock domains that is from the MAC clock domain to the application/system clock domain. Therefore, a minimum delay is required between the two consecutive timestamp captures. This delay is 4 clock cycles of MII and 3 clock cycles of PTP clocks. If the delay between the two captured timestamp is less than this delay, MAC does not take a timestamp snapshot for the second packet.

72.9.3.4 PTP processing and control

Table 479 shows the common message header for the PTP messages. This format is defined in the IEEE 1588-2008.

Table 479. Message format defined in IEEE 1588-2008

Bits								Octet	Offset
7	6	5	4	3	2	1	0		
transportSpecific				messageType				1	0
Reserved				versionPTP				1	1
messageLength								2	2
domainNumber								1	4
Reserved								1	5
flagField								2	6
correctionField								8	8
Reserved								4	16
sourcePortIdentity								10	20
sequenceId								2	30
controlField (*)								1	32
logMessageInterval								1	33

(*) – control Field is used in version 1. In version 2 message type field is used for detecting different message types.

There are some fields in the Ethernet payload that detects the PTP packet type and controls the snapshot to be taken. These fields are specified in Table 480 for PTP packets over Ethernet.

Following table provides the information about the fields that match to control the snapshots for the PTP packets sent over Ethernet for IEEE 1588 version 1 and version 2. The octet positions are offset by 4 for the tagged packets. This is based on the IEEE 1588-2008, Annex D, and the message format.

Table 480. Ethernet PTP packet fields required for control And status

Field matched	Octet position	Matched value	Description
MAC destination multicast address[1]	0–5	01-1B-19-00-00-00 01-80-C2-00-00-0E	All PTP messages can use any of the following multicast addresses[2]: • 01-1B-19-00-00-00 • 01-80-C2-00-00-0E[3]
MAC packet type	12, 13	0x88F7	PTP Ethernet packet
PTP control field (IEEE 1588 version 1)	46	0x00, 0x01, 0x02, 0x03, or 0x04	• 0x00: SYNC • 0x01: Delay_Req • 0x02: Follow_Up • 0x03: Delay_Resp • 0x04: Management
PTP message type field (IEEE 1588 version 2)	14 (nibble)	0x0, 0x1, 0x2, 0x3, 0x8, 0x9, 0xB, 0xC, or 0xD	• 0x0: SYNC • 0x1: Delay_Req • 0x2: Pdelay_Req • 0x3: Pdelay_Resp • 0x8: Follow_Up • 0x9: Delay_Resp • 0xA: Pdelay_Resp_Follow_Up • 0xB: Announce • 0xC: Signaling • 0xD: Management
PTP version	15 (nibble)	0x1 or 0x2	• 0x1: Supports PTP version 1 • 0x2: Supports PTP version 2

1. The unicast address match of destination addresses (DA), which is programmed in MAC address 0 to 31, is used if MAC_Timestamp_Control[TSENMACADDR] = 1.
2. IEEE 1588-2008, Annex F
3. MAC does not consider the PTP version 1 messages with peer delay multicast address (01-80-C2-00-00-0E) as valid PTP messages.

72.9.4 Transmit path functions

MAC captures a timestamp when the start packet delimiter (SFD) of a packet is sent on the MII interface. You can control the packets, for which the timestamps are captured on a per-packet basis and mark each transmit packet to indicate whether to capture a timestamp for it. MAC does not process the transmitted packets to identify the PTP packets. You can use the control bits in the transmit descriptor to specify the packets. MAC returns the timestamp to the software inside the corresponding transmit descriptor and therefore connects the timestamp automatically to the specific PTP packet. You must write the 64-bit timestamp information to TDES0 and TDES1 fields. The TDES0 field holds the 32 least significant bits of the timestamp.

72.9.5 Receive path functions

You can program MAC to capture the timestamp of all packets received on the MII interface or to process the packets to identify the valid PTP messages.

You can use these options of MAC_Timestamp_Control to control the snapshot of the time sent to the application:

- Enable snapshot for all packets
- Enable snapshot for IEEE 1588 version 1 or version 2 timestamp
- Enable snapshot for PTP packets transmitted directly over Ethernet
- Enable timestamp snapshot for the received packet for IPv4 or IPv6
- Enable timestamp snapshot only for EVENT messages (SYNC, DELAY_REQ, PDELAY_REQ, or PDELAY_RESP)
- Enable the node to be a master or slave and select the snapshot type

Table 481. Timestamp snapshot dependency on register bits

SNAPTYPSEL	TSMSTRENA	TSEVNTENA	PTP messages
00	x	0	SYNC, Follow_Up, Delay_Req, Delay_Resp
00	0	1	SYNC
00	1	1	Delay_Req
01	x	0	SYNC, Follow_Up, Delay_Req, Delay_Resp, Pdelay_Req, Pdelay_Resp, Pdelay_Resp_Follow_Up
01	0	1	SYNC, Pdelay_Req, Pdelay_Resp
01	1	1	Delay_Req, Pdelay_Req, Pdelay_Resp
10	x	x	SYNC, Delay_Req
11	x	x	Pdelay_Req, Pdelay_Resp

DMA returns the timestamp to the software inside the corresponding receive descriptor. The extended status, contains the timestamp message status and the IPC status. You must write the extended status in the normal descriptor RDES1 and the snapshot of the timestamp in RDES0 and RDES1 fields of context descriptor. The RDES0 field holds the 32 least significant bits of the timestamp.

See the System time correction for programming guidelines for IEEE 1588 timestamping (System time correction).

72.9.6 IEEE 1588 system time source

IP supports both the external and internal time source for reference timestamping.

- External time source uses the 64-bit external time reference and clock as an input in IP. The clock input synchronizes the external timing reference into the MAC clock domain. Upper 32-bit indicates the time in seconds and the lower 32-bit indicates the time in nanoseconds.
- Internal time source uses the only clock input to generate timing reference internally for snapshot and capture timestamps. Internal time reference has two fields.
  - UInteger48 seconds field: The seconds field is the integer portion of the timestamp in seconds units. It is 48-bit wide.

    For example, 2.000000001 seconds are represented as seconds field = 0x0000_0000_0002.

— UInteger32 nanoseconds field The nanoseconds field is the fractional portion of the timestamp in nanoseconds units.

For example, 2.000000001 seconds are represented as nanoseconds field = 0x0000_0001. The nanoseconds field supports the following two modes:

- Digital rollover mode: In this mode, the maximum value in the nanoseconds field is 0x3B9A_C9FF, that is, (10e9-1) nanoseconds.
- Binary rollover mode: In this mode, the nanoseconds field rolls over and increments the seconds field after 0x7FFF_FFFF value . Accuracy is ~0.466 ns per bit.

There is a system time register module, it is used when you use an internal time reference. The 80-bit time is maintained in this module and updated using the input reference clock (clk_ptp_ref_i). This time is the source for taking snapshots (timestamps) of Ethernet packets which is transmitted or received at the MII interface.

Initialize or correct the system time counter using the coarse correction method. In this method, write the initial value or the offset value to the timestamp update register. For initialization, write the system time counter with the value in the timestamp update register. For system time correction, the offset value is added to or subtracted from the system time.

In the fine correction method, correct the frequency offset and/or frequency drift of a slave clock (clk_ptp_ref_i) with respect to the master clock (as defined in IEEE 1588-2002) over a period of time instead of in one clock, as in coarse correction. This maintains linear time and does not introduce drastic changes (or a large jitter) in the reference time between the PTP sync message intervals.

See Initialization guidelines for system time generation for programming guidelines for IEEE 1588 timestamping (For internal timestamp source configuration).

72.9.7 IEEE 1588 higher word register

You can invoke the higher word register if system time source is internal. MAC timestamp is 64-bit wide. The values of the upper 16-bits of the seconds field are read from the CSR register.

72.9.8 Flexible pulse-per-second output

IP supports the flexibility of programming the start or stop time, generates pulse width and interval on pulse-per-second output. It also support four such output signals. By default, IP is in fixed pulse-per-second output mode with interval of 1 second.

Initially you must program the start time in target time registers "MAC_PPSx_Target_Time_Seconds and MAC_PPSx_Target_Time_Nanoseconds" for all desired or enabled pps output. If you re-programs start or stop time then you must do it after the synchronization of earlier programmed value. Bit 31 of MAC_PPS#_Target_Time_Nanoseconds register indicates that the synchronization is complete. If the application programs a start or stop time that has already elapsed, MAC sets an error status bit which indicates the programming error. If enabled, MAC also sets the target time reached interrupt event. The application can cancel the start or stop request only if the corresponding start or stop time has not elapsed. If the time has elapsed, the cancel command has no effect.

Program the PPS width and interval in terms of granularity of system time, that is, the number of units of sub-second increment value.

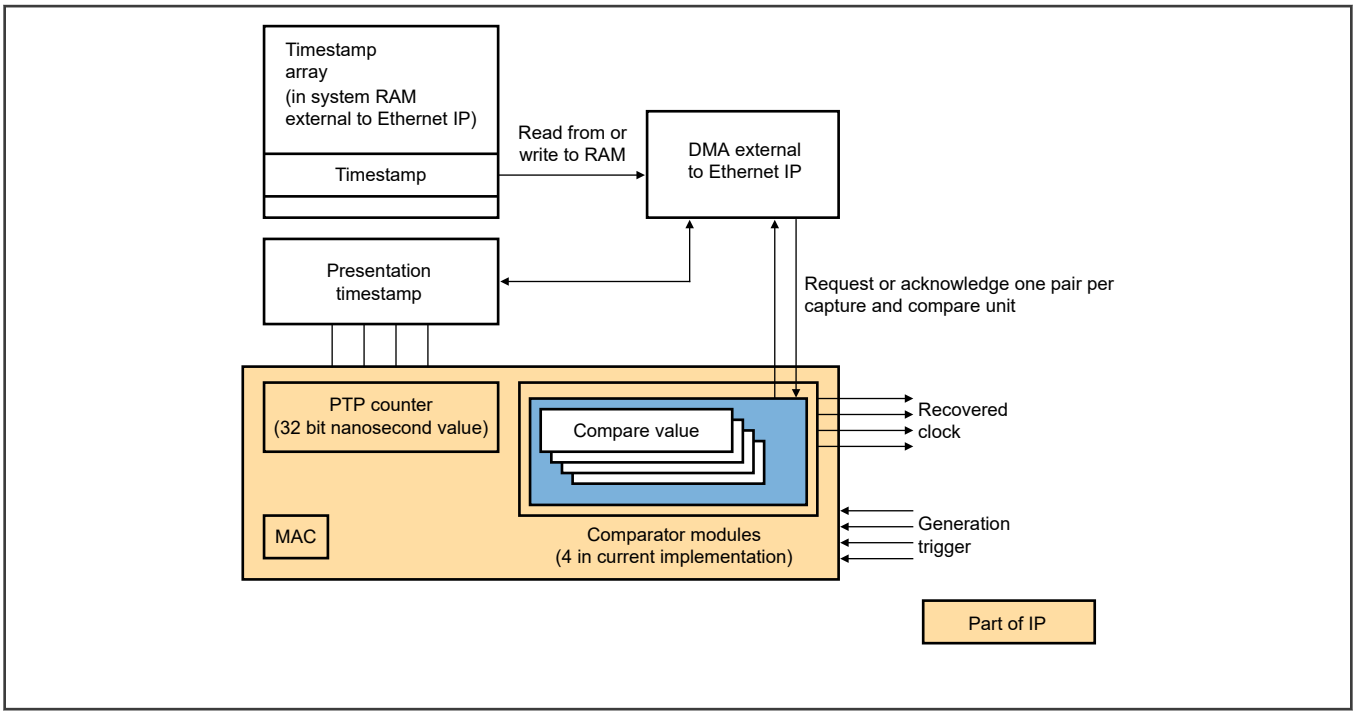
See Programming guidelines for flexible pulse-per-second output for more information.

72.9.9 Media clock generation and recovery

Media clock recovery and generation requires a dedicated hardware and software to complete the operation. IP generates a reference clock with the help of a dedicated hardware which multiplies to get the desired media clock. Software extracts the presentation time from the incoming PTP 1722 frames and program these values into CSR registers so that IP can read them for media clock recovery.

Figure 396 shows that the timestamp array and the DMA module are external to IP. IP supports a 32-bit presentation time counter in nanoseconds that completes at the full 32-bit value to match with 1722 presentation time format and an handshake mechanism with an external application (such as DMA) to program the presentation time into CSR register.

Figure 396. Media clock generation and recovery block diagram



72.9.9.1 Presentation time counter

Presentation time counter provides an another perspective of the PTP system time (1588 timer). For a given PTP system time (PTP[63:32] represents time in seconds Comparator modules and PTP[31:0] represents time in nanoseconds), there exists a corresponding presentation time (32-bit value in nanosecond). The presentation time computed is referred to as current presentation time (CPT).

CPT derives from 64-bit PTP system time. PTPNS[63:0] = (PTP[63:32] * 32'd1,000,000,000) + PTP[31:0] where, PTPNS is the PTP system time converted into a 64-bit nanosecond value.

Current presentation time[31:0] = PTPNS[31:0]

Media clock recovery is possible when the system time is internally generated in IP and compute the CPT in the same way as the internal PTP system time generation. The increment cycle of CPT and system time are same because the values of both the timers are in nanoseconds and they are synchronous. The timer updates at the same instance, but the updated value could be different. You must compute a separate 32-bit update value in nanoseconds for the CPT update. CPT sampled at different edges of a triggering input generates media clock timestamps that are inserted in 1722 based AVBTP packets, to recover the clock at the destination.

72.9.9.2 Comparator modules

Figure 396 shows that the module supports four comparator modules to handle multiple media clocks that it may require. When MAC_Timestamp_Control[PTGE] is 1, it indicates that the comparator modules handshake with the application to program the timestamps into the MAC_PPS(#i)_Target_Time_Seconds register.

Write 1 to presentation time control fields of the particular instance for recovery mode.

The timestamps received from the application are referred as target presentation time (TPT).

When MCGREN#i field of MAC_PPS_Control is 1, it indicates that the IP operates in MCGR mode. The comparator module sends upto two requests to the application for TPT write, when MAC_Timestamp_Control[PTGE] and the presentation time control bits of the particular instance are 1 for MCGR mode. Subsequent requests are generated each time a presentation time match occurs. CPT transitions from a value less than TPT, to a value greater than or equal to CPT. The first request asserts when a specific comparator instance sets to MCGR mode with a non-zero presentation time control and an additional request is made when the comparator receives the first data. This allows the application to write the next TPT value when IP processes the previous TPT value for a match.

TPT read from the MAC_PPS(#i)_Target_Time_Second is considered as a future time. A toggle or pulse is generated in the next cycle when a match is detected. Also,the mcgr_dma_req_o#i request for that comparator asserts (to obtain next TPT) until the corresponding application acknowledgment is set, when a match is detected.

The presentation control fields (PPSCMD#i field of MAC_PPS_Control) determine the shape of the generated waveform. You can program these fields to either toggle or generate a high/low pulse for one PTP clock cycle, when matched.

Media clock recovery: A match occurs, when the free-running CPT value matches the received TPT value. An output signal, EMAC_TMR signal as shown in the IO mux sheet asserts (toggle, low pulse, or high pulse) on the basis of the programmed presentation control value.

Media clock generation: On the basis of the presentation control value programmed in MAC_PPS_Control, the particular comparator captures the presentation time and programs it into MAC_PPS(#i)_Target_Time_Second register. The captured timestamp is read when a request is raised to the application. No new timestamps are captured, until the read operation is complete acknowledgment is received from the application.

72.9.9.3 Media clock generation and recovery flow

Figure 397 shows the media clock generation and recovery flow. Write 1 to MAC_Timestamp_Control[PTGE] for CPT generation and write 1 to MAC_PPS_Control MCGREN#i field to enable the corresponding instance in the MCGR mode, for both media clock generation and recovery.

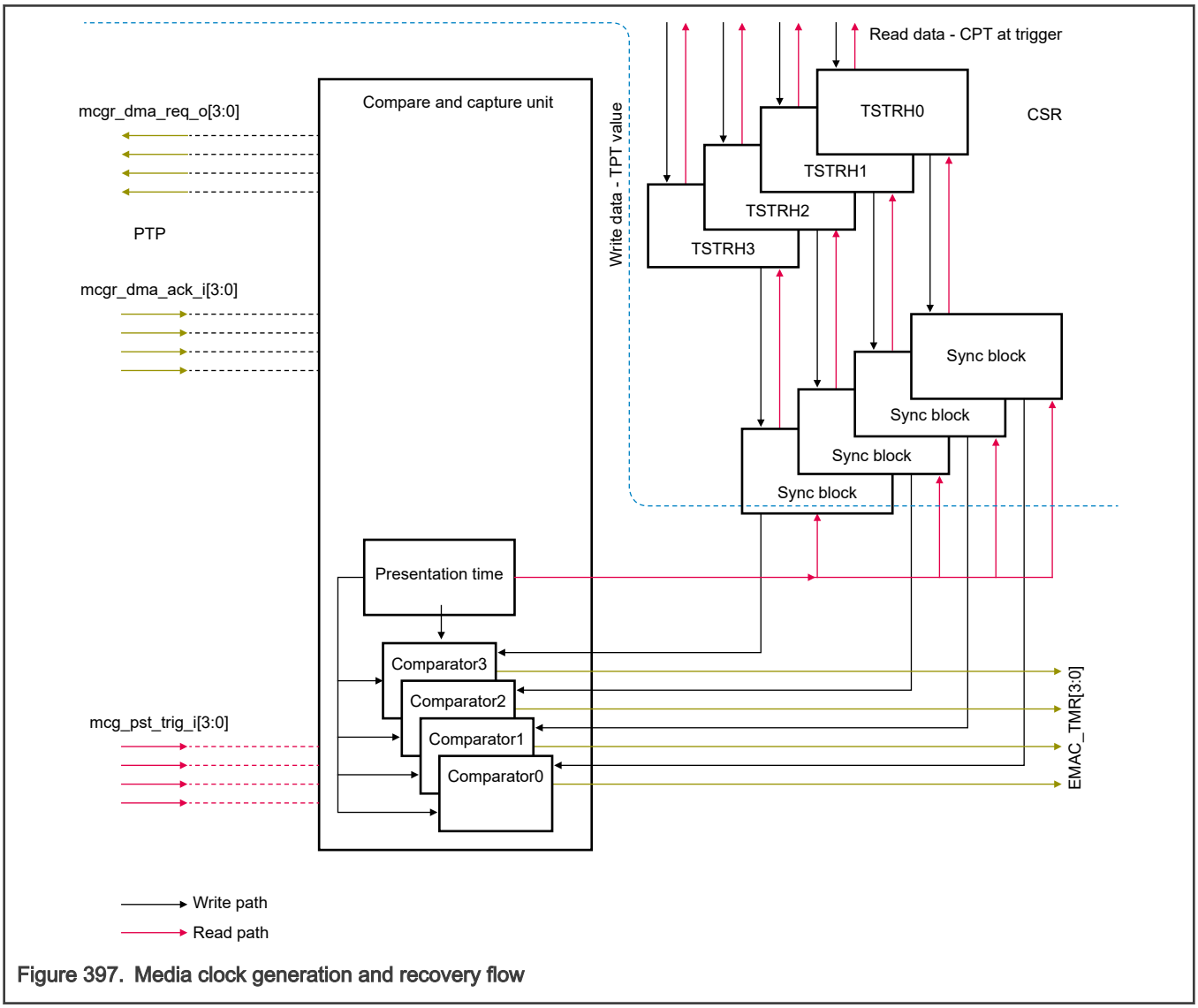


Figure 397. Media clock generation and recovery flow

NOTE

1. The consecutive triggers to sample the presentation time must assert after a few PTP clock cycles so as to allow synchronization delays. (There is no such issue when the input trigger maximum frequency is 8 KHz and the PTP clock runs at least at 1MHz)
2. In the Media Clock Recovery mode, the DMA acknowledgment is both posted as well as non-posted.

See Programming guidelines for media clock generation and recovery for more information.

72.9.10 One step timestamp

IP supports the one step timestamp feature. Writing 1 to bit 20 (OSTC) in the control word, enables the one step timestamp feature for a packet.

See the Programming guidelines for IEEE 1588 timestamping for more information.

72.9.11 IEEE 1588 sub nanoseconds timestamp

IP supports the ingress and egress correction accuracy in sub nanoseconds. It is programmed in MAC_Timestamp_Ingress_Corr_Subnanosec[TSICSNS] and MAC_Timestamp_Egress_Corr_Subnanosec[TSECSNS], respectively. In this case, the correction has an unit of nanosecond, multiplied by 2^8. You must program the least significant 8 bits of the value in the sub-nanoseconds register.

For example, if the required egress correction is 3.6 nS and MAC_Timestamp_Control[TSCTRLSSR] is 1, then MAC_Timestamp_Egress_Corr_Nanosecond[TSEC] must be 0x3 and MAC_Timestamp_Egress_Corr_Subnanosec[TSECSNS] must be 0x99 (0.6 * 256). Similarly, if the required ingress correction is -3.6 nS and MAC_Timestamp_Control[TSCTRLSSR] is 1, then MAC_Timestamp_Ingress_Corr_Nanosecond[TSIC] must be 0xBB9A_C9FC and MAC_Timestamp_Ingress_Corr_Subnanosec[TSICSNS] must be 0x66 (fractional part of (10^9 - 3.6) * 256).

72.10 Multiple channels and queues support

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

IP support two queues and channel on Tx and Rx paths. Figure 398 shows the architecture of IP with two queues and channel.

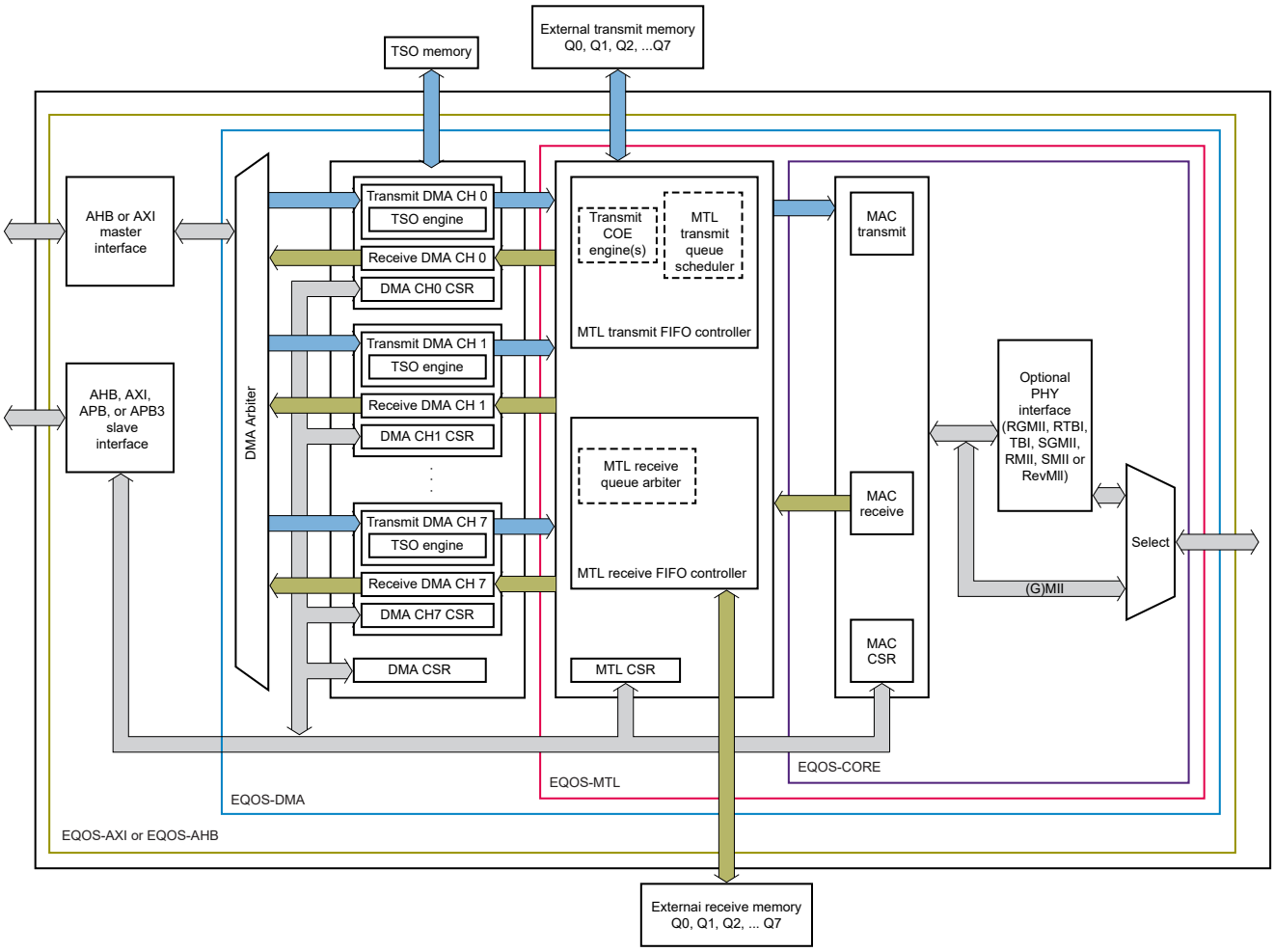


Figure 398. Module with multiple channels and queues block diagram

The above block diagram, support only two queues.

IP DMA arbiter support two types of arbitration scheme, fixed priority and weighted round-robin. The DMA arbiter performs the arbitration between the Tx and Rx paths of DMA channels to access descriptors and data buffers. DMA_Mode[DA] specifies the arbitration scheme (fixed or weighted round-robin) between the channel Tx and Rx DMA.

DMA_Mode[TXPR] sets the priority between the corresponding Tx DMA and Rx DMA. Writing 1 to DMA_Mode[PR] specifies the weighted priority between the Tx DMA and Rx DMA in round robin arbitration scheme.

Table 482 provides an information about the priority scheme between Tx DMA and Rx DMA.

Table 482. Priority scheme for Tx DMA and Rx DMA

Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Priority schemes
x	x	x	0	1	Rx always has priority over Tx
0	0	0	0	0	Tx and Rx have equal priority. Rx gets the access first on simultaneous requests
0	0	1	0	0	Rx has priority over Tx in ratio 2:1

Table continues on the next page...

Table 482. Priority scheme for Tx DMA and Rx DMA (continued)

Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Priority schemes
0	1	0	0	0	Rx has priority over Tx in ratio 3:1
0	1	1	0	0	Rx has priority over Tx in ratio 4:1
1	0	0	0	0	Rx has priority over Tx in ratio 5:1
1	0	1	0	0	Rx has priority over Tx in ratio 6:1
1	1	0	0	0	Rx has priority over Tx in ratio 7:1
1	1	1	0	0	Rx has priority over Tx in ratio 8:1
x	x	x	1	1	Tx always has priority over Rx
0	0	0	1	0	Tx and Rx have equal priority. Tx gets the access first on simultaneous requests
0	0	1	1	0	Tx has priority over Rx in ratio 2:1
0	1	0	1	0	Tx has priority over Rx in ratio 3:1
0	1	1	1	0	Tx has priority over Rx in ratio 4:1
1	0	0	1	0	Tx has priority over Rx in ratio 5:1
1	0	1	1	0	Tx has priority over Rx in ratio 6:1
1	1	0	1	0	Tx has priority over Rx in ratio 7:1
1	1	1	1	0	Tx has priority over Rx in ratio 8:1

72.10.1 Multiple queues and channels support in the transmit path

IP support two transmit queues and two channels.

Fixed priority scheme is the default priority scheme for the DMA channels and channel with highest priority always wins the arbitration when it requests for the bus. Channel 0 is always of lowest priority and channel 1 is always of highest priority.

In weighted strict priority (WSP), the weight corresponds to the number of burst transfers given to a channel in one arbitration cycle. Reallocate the unused burst transfers of one or more channels on the basis of the priority. The channel with highest priority receives the unused burst transfer time before it is allocated to a channel with the next highest priority. You can process the next lower priority, when a channel uses the allocated burst transfers. After processing the allocated bandwidth of all the channels that have packets to transmit, allocate any unused burst transfer time to the channel of the highest priority (if required), and then next highest priority (if required), and so on.

In weighted round robin (WRR) priority scheme, program the weight through TCW field of DMA_CH#_Tx_Control register. IN WRR scheme, service all channels in round-robin order according to the weights settings. Table 483 shows that the TCW field of DMA_CH#_Tx_Control register provides the weight for each transmit channel.

Table 483. Weight for DMA channels

TCW Field	Transmit Channel Weight
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

The configured weights correspond to the number of burst transfers given to a channel in one arbitration cycle. The unused or excess burst transfers are distributed equally to all channels.

See Programming guidelines for multi-channel multi-queuing for more information.

72.10.2 Multiple queues and channels support in the receive path

IP support two receive channels and two queues.

In the receive direction, the MTL Rx controller selects the Rx DMA for which it transfers or reads the data from the receive FIFO memory. This scheduling is based on the programming done in the respective MTL_RxQ[x]_Control register.

Each Rx DMA indicates when it is ready to transfer data and the size of the burst-length (number of beats) that it will transfer. The scheduler checks whether sufficient data (of requested burst length) is available to transfer to these DMAs and then selects the receive DMA that is serviced using the programmed priorities.

See Programming guidelines for multi-channel multi-queuing for more information.

72.10.3 Rx queue to DMA mapping

You can program MTL_RxQ_DMA_Map0 (for queues 0 and 1) to route the packets in the MTL receive queues to any one of the multiple DMA channels.

The following types of receive queue to DMA mapping is possible through programming.

72.10.3.1 Static mapping

In this mode, you can connect all the packets of the receive queue to a specific DMA channel. For example, all the packets from the receive queue 0 program MTL_RxQ_DMA_Map0[Q0MDMACH] (bit[3:0]) and MTL_RxQ_DMA_Map0[Q0DDMACH] (bit[7] = 0) to route to a DMA channel.

Similarly, packets from other receive queues program register fields corresponding to each queue to route to any DMA channel.

72.10.3.2 Dynamic (per packet) mapping

In this mode, the destination DMA channel of a packet read from a Rx queue is not constant but it is decided independently for each packet. For example, if you write 1 to MTL_RxQ_DMA_Map0[Q1DDMACH], it indicates that the static mapping disables for receive queue 1 and ignores the value in MTL_RxQ_DMA_Map0[Q1MDMACH]. The MAC receiver decides the destination DMA channel for each packet, depending on the following in decreasing order of priority:

1. **L3-L4 filter based DMA selection**: The TCP/UDP and IP header fields of the received packet are matched against the corresponding values programmed and enabled for comparison in the MAC_L3_L4_Address_Control register. If the match is successful, the DMA channel number programmed in the DMCHN field of the MAC_L3_L4_Address_Control register is selected as the destination DMA channel number, if DMCHEN field of the same register is 1. If none of the L3-L4 registers give a comparison match, then the module proceeds to the next step below.
2. **Extended VLAN based DMA selection**: Extended routing is applicable only if the VLAN filter has passed. Routing is done only on the basis of the perfect filter result. Each perfect filter has a DMA channel enable and a DMA channel number field which you can program. Routing is applicable for a filter, only if the DMA channel enable field is 1. The frame routes to the smallest matching filter's DMA channel, if it is enabled. If the filter's DMA channel number is not enabled, the frame route to channel 0. For example, if a frame's VLAN tag matches filters 7, 3, and 1, then the MAC checks if DMA channel number of filter 1 is enabled through programming. If yes, the frame route to the programmed value; otherwise it route to DMA. When the inverse filter is enabled; is routed to the least mismatched filter's DMA channel number provided it is enabled. If the DMA Channel enable bit is not set, then the frame routes on the basis of the DA based addressing or to channel 0.

   If hash filter is also enabled, it is determines the filter result only. Routing will still depends on the enabled perfect filters. Routing based on the VLAN filter will not occur if none of the perfect filters are enabled or if all of them are bypassed. The frame will route via DA based addressing or to channel 0. If all the perfect filters fails and the hash filter has passed, then the VLAN filter result is a pass but routing will be based on DA based addressing or to channel 0. Similar behavior is seen when you enables inverse filtering as well.
3. **Ethernet DA-based DMA selection**: The DA address of the received packet is compared against the programmed DA values in MAC address registers. If the address matches any of the programmed values, the corresponding DCS field (when enabled) determines the destination DMA channel number.

If none of the above operations make a successful match or decision, then the packet routes to the DMA channel 0 by default.

72.10.3.2.1 Broadcast/multicast packet duplication

This feature provides a mechanism to send the received broadcast or multicast packets to multiple DMA channels.

MAC_Address0_High[DCS] and other optional MAC_Address(#i)_High (i=1 to 2) registers determines the DMA channel number to which the received packet (that matches the MAC address present in that register) must route.

DMA channel routing mechanisms such as extended VLAN, or L3-L4 based routing does not support the packet duplication.

The packet duplication feature is supported only on the highest MTL queue configured. Therefore, you must program MAC_RxQ_Ctrl1[MCBCQ] to the highest RxQ present in the configuration for the packet duplication for broadcast or multicast packets.

72.10.4 Selection of tag priorities assigned to receive queues

Program the PSRQ field in the corresponding MAC_RxQ_Ctrl2 to assign the VLAN tag priorities to receive queues. The bit corresponding to the VLAN tag priority can be set in the PSRQ field for assigning that priority to the receive queue. For example, if you want to assign VLAN tag priority of 3 to receive queue 0, write 1 to bit [3] in PSRQ field of MAC_RxQ_Ctrl2. The VLAN tag priority assigned to particular receive queue must be unique, that is, you cannot assign more than one receive queue to the same VLAN tag priority. However, more than one VLAN tag priorities can be assigned to same receive queue.

The settings in the PSRQ field is used for VLAN tagged receive packet routing to receive queues as well as for PFC based transit flow control. The received VLAN tagged receive packet route to receive queue that has the VLAN tag priority match. In PFC based transit flow control, PSRQ field corresponding to a particular receive queue enables VLAN tag priorities in the PFC packet transmitted when corresponding receive queue reach the threshold level.

72.10.5 Receive side routing from MAC to queues

IP supports receive side routing from MAC to queues. MAC route the receive packets to the receive queues on the basis of the these packet types in that order

- Unicast/Multicast destination address packets that fail the destination address filter

- Multicast/Broadcast destination address packets that pass the destination address filter
- VLAN tag priority field in VLAN tagged AV data packets
- AV control packets
- VLAN tagged IEEE 1588 PTP over Ethernet packets
- Untagged IEEE1588 PTP over Ethernet packets
- VLAN tag priority field in VLAN tagged packets
- Untagged packets

You can route the outer C-VLAN tagged AV data receive packets on the basis of the priorities assigned to receive queues through PSRQ field in the corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 and the corresponding receive queue is enabled for AV through RXQ#EN field in MAC_RxQ_Ctrl0. These packets may be single VLAN tagged with C-VLAN type or double VLAN tagged with outer VLAN tag of C-VLAN type when double VLAN feature enables (MAC_VLAN_Tag_Ctrl[EDVLP] = 1) with an inner C-VLAN tagged or inner S-VLAN tagged when SVLAN processing is enabled (MAC_VLAN_Tag_Ctrl[ESVL] = 1). This type of Rx packet routing is available when you select the AV feature and multiple receive queues in the configuration.

You can route the AV control receive packets on the basis of the receive queue number specified in MAC_RxQ_Ctrl1[AVCPQ] and the corresponding receive queue is enabled for AV through RXQ#EN field in MAC_RxQ_Ctrl0. These packets may be single VLAN tagged with C-VLAN type or double VLAN tagged with outer VLAN tag of C-VLAN type when double VLAN feature is enabled (MAC_VLAN_Tag_Ctrl[EDVLP] = 1) with inner C-VLAN tagged or inner S-VLAN tagged when SVLAN processing is enabled (MAC_VLAN_Tag_Ctrl[ESVL] = 1). This type of receive packet routing is available when you select the AV feature and multiple receive queues in the configuration.

You can route the VLAN tagged receive packets on the basis of the priorities assigned to receive queues through PSRQ field in corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 and RXQ#EN field in MAC_RxQ_Ctrl0 enables the corresponding receive queue for DCB/generic. This type of receive packet routing is available when you select the multiple receive queues in the configuration.

You can route the untagged IEEE 1588 PTP over Ethernet receive packets on the basis of the receive queue number specified in MAC_RxQ_Ctrl1[PTPQ] and the corresponding receive queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0. Also, you can route the VLAN tagged IEEE 1588 PTP over Ethernet receive packets on the basis of the priorities assigned to receive queues through PSRQ field in corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 or the receive queue number specified in the MAC_RxQ_Ctrl1[PTPQ] and RXQ#EN field in MAC_RxQ_Ctrl0 register enables the corresponding Rx queue. Programming MAC_RxQ_Ctrl1[TPQC] determines this. This type of receive packet routing is available when IEEE 1588 timestamp feature supports and the multiple receive queues are selected in the configuration. The VLAN tagged IEEE 1588 PTP over Ethernet receive packets are detected only when you disables 802.1AS mode (MAC_Timestamp_Control[AV8021ASMEN] = 0), otherwise this type of receive packets are routed as generic VLAN tagged Rx packets.

You can route the multicast or broadcast destination address receive packets that passes the destination address filter on the basis of the receive queue number specified in MAC_RxQ_Ctrl1[MCBCQ]the MCBCQ field of MAC_RxQ_Ctrl1 register when enabled through MAC_RxQ_Ctrl1[MCBCQEN] and the corresponding receive queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0. This type of Rx packet routing is available when you select the multiple receive queues in the configuration.

You can route the untagged receive packets on the basis of the receive queue number specified in MAC_RxQ_Ctrl1[UPQ] and RXQ#EN field in MAC_RxQ_Ctrl0 enables the corresponding receive queue. This type of receive packet routing is available when you select the multiple receive queues in the configuration.

The unicast destination address receive packets that fail the destination address filter can be routed based on the receive queue number specified in MAC_RxQ_Ctrl4[UFFQ] when enabled through MAC_RxQ_Ctrl4[UFFQE]UFFQE, MAC_Packet_Filter[RA] = 1 and the corresponding receive queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0. This type of receive packet routing is available when you select the multiple receive queues in the configuration.

You can route the multicast destination address receive packets that fails the destination address filter on the basis of the receive queue number specified in MAC_RxQ_Ctrl4[MFFQ] when MAC_RxQ_Ctrl4[MFFQE] enables it. MAC_Packet_Filter[RA] = 1 and RXQ#EN field in MAC_RxQ_Ctrl0 enables the corresponding receive queue. This type of receive packet routing is available when you select the multiple receive queues in the configuration.

The receive packet will route through the receive queue 0, if it is not classified in any of the defined packet type for routing.

72.10.6 Receive side arbitration between DMA and MTL

IP controller supports receive side arbitration between DMA and MTL.

After the current packet processing completes, the DMA channel controller fetches the next descriptor and after the descriptor fetching completes, the DMA channel controller evaluates the amount of data to transfer to the Rx buffer on the basis of the programmed PBL and receive buffer length. Accordingly, it requests the MTL to transfer the data.

After servicing the current request, the MTL receive queue arbitration scheme selects receive queue on the basis of the arbitration scheme (MTL_Operation_Mode[RAA]) and the weights programmed in queue <n> receive control register. The arbitration is done among queues for which DMA is ready to service. After the receive queue is selected, PBL amount of data is read out from that queue and route to the receive DMA channel on the basis of the receive channel selection criteria.

The arbitration occur after every PBL data transfer completes and descriptors are ready for processing from at least one DMA channel.

72.10.7 Transit side arbitration between DMA and MTL

IP supports transit side arbitration between DMA and MTL. Transit DMA channels and transit queues are always mapped directly because the number of transit DMA channels are always equal to the number of transit queues in MTL. For instance, each transit DMA pushes data into its respective transit queue assigned to it.

The data inside each transit queue is stored in packets. Therefore, if two DMAs are allowed to transfer data into the same queue, when a transit DMA starts a packet transfer, the other DMA cannot transfer data unless the previous packet is completely pushed-in. This means that the second DMA remains idle until the first packet is transferred. Hence, each DMA is always connected directly to its corresponding transit queue.

72.10.8 Audio video bridging

IP supports the audio video bridging in 100 Mbps mode only, which allows the transmission of time sensitive traffic over network. IP implements these protocol for supporting the AVB feature:

- IEEE 802.1Qbv-2015 (Enhancements to scheduling traffic)
- IEEE 802.3br (Interspersing traffic)
- IEEE 802.1Qbu (Frame preemption)

The queue 0 transmit path supports the strict priority algorithm, and it is used for best-effort traffic when transmit paths of additional queues use the credit-based shaper algorithm to support traffic management. For a queue, the credit-based shaper algorithm determines that a queue is available for transmission if these conditions are true:

- The queue contains one or more packets.
- The credit for the queue is positive per the algorithm
- AV traffic is received on all queues. IP can also receive untagged PTP packets in addition to AV traffic on any queue.

72.10.9 Queue modes

Transit and receive queues both handles the AV traffic.

Queueing is based on WRR (Weighted round robin) or WSP (Weighted strict priority) algorithm for generic traffic. It is based on CBS (Credit Base Shaper) and SP (Strict Priority) algorithm for AV traffic.

The receive queue is configured for generic or AV based routing, which depends on the RXQ#EN field of corresponding queue. Queueing is done on the basis of the VLAN tag priority. The VLAN tag priority must match the PSRQ field of MAC_RxQ_Ctrl2. The receive packets can route to a particular DMA channel on the basis of the DCS field of perfectly-matched MAC address register.

By default, the untagged packets in a generic traffic route to the receive queue specified in MAC_RxQ_Ctrl1[UPQ]. Queue 0 is the default value of MAC_RxQ_Ctrl1[UPQ]. You can override the default value with any other value, for the MAC_RxQ_Ctrl1[UPQ].

The AV control packets (tagged or untagged) route on the basis of MAC_RxQ_Ctrl1[AVCPQ]. The PTP over Ethernet packets route on the basis of MAC_RxQ_Ctrl1[PTPQ].

72.10.10 Queue priorities

You can program the priority of an receive queue in the corresponding field of MAC_RxQ_Ctrl2. Also, you must program the AV queue (high priority) as an higher priority than the best-effort queue (low priority).

72.10.11 Enhancements to scheduled traffic (EST)

The IEEE 802.1Qbv-2015 defines the schedule for each queues on every egress port which makes the implementation aware of traffic arrival schedule. This information blocks the lower priority traffic from transmission in this time window or slot. This ensures that the sender forward the scheduled traffic to receiver through all the network nodes with a deterministic delay.

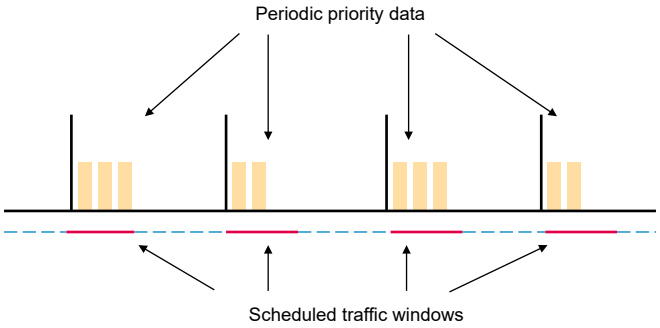


Figure 399. Time aware shaper implementation for traffic scheduling

An important requirement to achieve a low latency is to ensure that there are no interfering frames during the scheduled windows that are reserved for high priority traffic. The use of scheduled traffic imposes limitations when a transmission starts.

As shown in Figure 400 , if an interfering frame begins transmission just before the reserved time period starts, it can extend transmission into the reserved window, and potentially interfere with higher priority traffic. Therefore, a guard band whose size is equal to the largest possible interfering frame is required before the window starts.

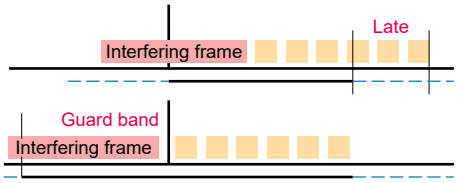


Figure 400. Implementing a guard band to avoid delays due to interfering frames

A larger guard band equates to a less efficient use of network bandwidth. However, this issue is addressed with the implementation of IEEE802.1Qbu (frame preemption). Frame preemption breaks the interfering frame into smaller fragments. Therefore, the guard band must be as large as the largest possible interfering fragment instead of the largest possible interfering frame.

During the guard band, only the frames that can complete the transmission of the entire frame before the next gate close event are permitted. This ensures that the high priority traffic can always start at the beginning of the window reserved for it.

72.10.11.1 Frequently used terms in EST

These are some of the frequently used terms in the EST support and their definitions.

- Gate control list: The list in the hardware memory that hold the gate controls and the associated time intervals.
- Gate controls: For a given schedule (row in gate control list) there is a gate open (O) and gate close (C) state associated with each TC. The set of O or C values, whose width is same as configured TCs is called the gate controls. Example: CCOCOOCO means TC7=C, TC6=C, TC5=O and so on.

- Time interval: Time interval (in nanoseconds) is a 16, 20, or 24-bit configurable field in the gate control list that indicates the time for which the associated gate controls are valid.
- Base time register: Each gate control list is associated with a 64-bit base time register that holds the start time (in PTP format) for the list.

72.10.11.2 Updates to the transmit scheduling to support EST

To support EST, these updates are required to transmit scheduling:

- Implementation of gate control list (GCL)
- Enforcing gate controls in the scheduler
- Utilizing gate open (O) duty cycle in the computation of idleSlope (CBS)

72.10.11.2.1 Implementation of gate control list (GCL)

Figure 401 shows how the gate control list governs the gate close(C) and open(O) events on the basis of the schedule provided for each event. GCL has two parts:

- Time interval: Defines the time in nanoseconds for which the gate controls are valid and must apply before reading the next gate controls from the list. The configurable width of 16, 20 or 24-bit represents a maximum of 64us, 1ms, or 16ms schedule interval respectively. It supports left shift of the time interval upto 7 bits to be able to apply a multiplication factor from 1 to 128ns (in steps of 2(power)n). The maximum value (post shifting) of this field must be 999,999,999 ns.
- Gate control: Defines the open (O) represented by logic 1 or close (C) represented by logic 0 state for the gate of each TC.

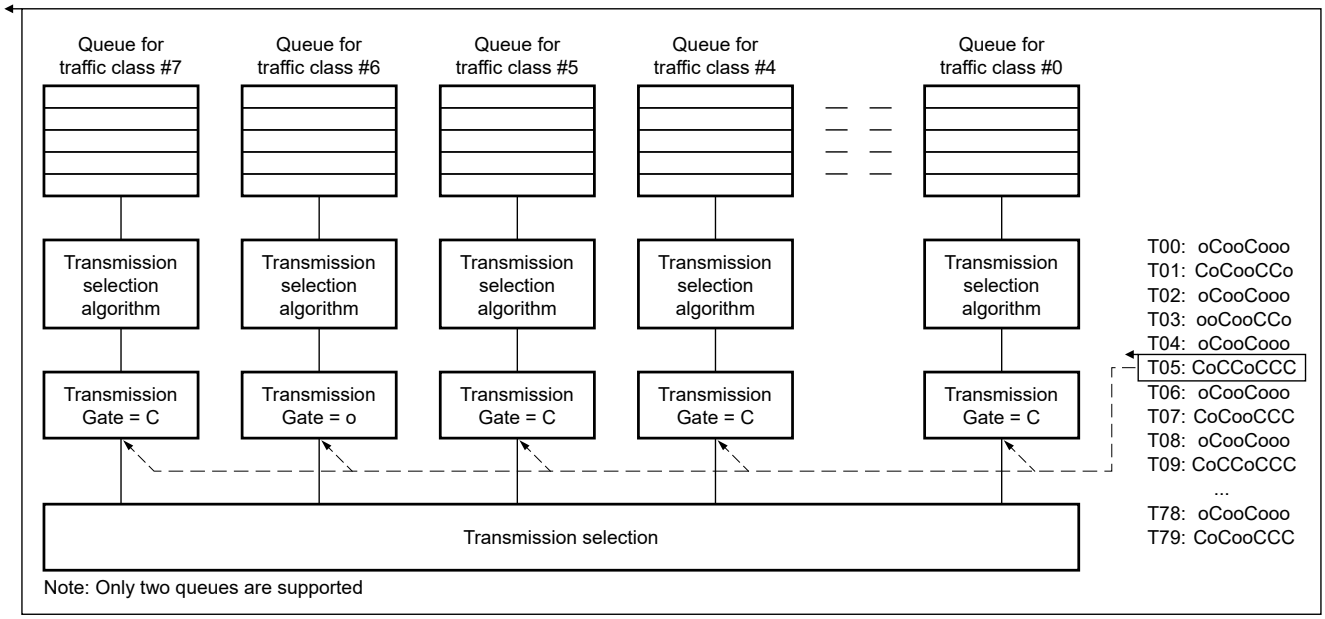


Figure 401. Block diagram from IEEE 802.1 Qbv specification - GCL governing gate close and open events

The implementation of GCL consists of these two gate control lists:

- HWOL - Hardware owned list which is a list for hardware access.
- SWOL - Software owned list which is a list for software access.

The access to these lists is mutually exclusive. Hardware sets the ownership to the list in MTL_EST_Status[SWOL]. Table 484 provides the implementation details of GCL.

Table 484. External memory used for holding the two gate-control lists

Gate control (up to 8 bits)	Time interval (ns) 16, 20, or 24 bits	
OOCCCCCC	T0	HWOL or SWOL
OOOOCCCC	T1	
CCCCOOCC	T2	
CCCCCCOO	T3	
I	I	
I	I	
OCOCOCOO		
OOOOCCCC	T last	
OOOOCCCC	T0	SWOL or HWOL
OOOOCCCC	T1	
OOCCCCCC	T2	
OOOOCCCC	T3	
I	I	
I	I	
OCOCCCCC		
OOCCCCCC	T last	

72.10.11.2.2 Registers related to gate control list

The following are the set of four registers (one for each GCL) related to GCL. These registers are implemented through Indirect addressing using MTL_EST_GCL_Control and MTL_EST_GCL_Data registers.

1. 64-bit Base time register (BTR)

2. 40-bit Cycle time register (CTR)

3. m-bit Time extension register (TER)

4. n-bit [Gate control] List length register (LLR) (n depends of the configured GCL depth)

72.10.11.2.2.1 Base time register (BTR)

This is a 64-bit register that specifies the start time to execute the GCL. The format of the BTR is same as the PTP format (upper 32-bits holds time in seconds and lower 32 bits hold time in nanoseconds). After the execution of a given list starts, the implementation can update the value in BTR to indicate the next list execution start time.

72.10.11.2.2.2 Cycle time register (CTR)

This is a 40-bit register that specifies the time at which the execution of the GCL must repeat. This register consists of an 8-bit value in seconds, and a 32-bit value in nanoseconds (similar to the PTP time format with truncated seconds register). For a given GCL, the start time is "Base Time" + N * "Cycle Time" where N is an integer value that represents the iteration number starting with 0 for first iteration. If the GCL execution takes longer than the cycle time, then the list is truncated at the cycle time and the subsequent loop begins at cycle time.

72.10.11.2.2.3 Time extension register (TER)

This is a m-bit (where m = Configured time interval width + 7) register that specifies the amount of time (in nano seconds) the current GCL can extend before switching to the new GCL. This helps to avoid the execution of the small fragments of the current list before switching to a new list.

72.10.11.2.2.4 List length register (LLR)

This is an n-bit register (when n is 7, 8, 9, 10, or 11 for a GCL configured depth of 64, 128, 256, 512, or 1024 respectively) that specifies the integer value of the length of the GCL (that is the number of valid rows in each GCL). The processing of the GCL stops after the number of rows read equals to the LLR value.

72.10.11.2.3 Transmission gating implementation

A bridge or an end station can enhance to allow transmission from each TC that is yet to be scheduled relative to a known timescale. To achieve this, a transmission gate is associated with each TC; the state of the transmission gate determines if you can select the queued frames for transmission.

For a TC, the transmission gate can be in one of these two states:

1. Open: Select queued frames for transmission, in accordance with the definition of the transmission selection algorithm associated with the TC.
2. Closed: Does not select queued frames for transmission.

A frame on a traffic class queue is not available for transmission if the transmission gate is in the closed state or if there is insufficient time available to transmit the entirety of that frame before the next gate-close event associated with that queue.

The implementation has visibility to the current schedule of gate controls and the immediate next schedule of the gate controls. So the maximum Gate Open period does not exceed the sum of the two Time Intervals. This is because, a frame is selected for transmission only if the gate is currently Open and the duration of gate open interval is greater than the time taken to transfer the entire frame.

The implementation must know the frame size before the transmission, so that you can avoid the transmission overruns and only the frames that can complete are scheduled at all times.

The implementation adequately compensates for the implementation delays in the data transfer from the buffer to the line by offsetting the current time with all the relevant delays (provided by MTL_EST_Control[CTOV]). This ensures that the schedule provided is always accurately implemented at the line.

You must ensure that the GCL slot interval is always greater than the expected packet size and overhead (scheduler delay, inter frame gap (IFG), and preamble, all combined).

72.10.11.3 Idle slope computation updates

When EST is enabled, credit accumulates only when the gate is open therefore, the effective data rate of the idleSlope must increase to reflect the duty cycle for the transmission gate associated with the queue.

The idleSlope is computed on the basis of the gate open time and oper cycle time values. Program the idleSlope registers (implemented one per CBS enabled TC) based on the following equation. The existing MTL register has sufficient field width to accommodate the new values for idleSlope.

idleSlope = (operIdleSlope(N) * OperCycle/GateOpenTime)

72.10.11.3.1 Operational details of GCL

Write 1 to MTL_EST_Control[SSWL], so that the hardware can access the programmed gate control list. The first set of gate controls are applied when the current time is equal to the value in the base time register (BTR) and is held until the programmed "time interval" value.

One additional gate control event is always read ahead from the list, to avoid the transmission overruns. This enables the GCL to determine the next gate close events (if any) for the open TCs.

The scheduling is done on the basis of the gate open state and time interval of only the current and subsequent schedule. An internal accumulator adds the time intervals when gate controls are applied. BTR + Accumulator specifies the time at which the next set of gate controls are applied.

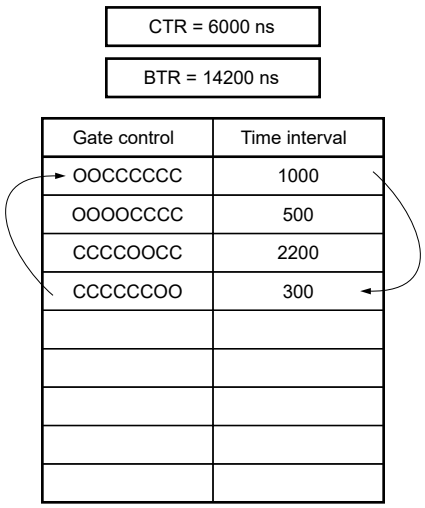


Figure 402. GCL and associated registers - basetime and cycletime

GCL is read progressively from the first row adhering to the schedule. The read operations continue until the list length (from LLR register) is reached and the execution of the list restarts at BTR + CTR time. At this point the value in CTR increments BTR to mark the beginning of a new cycle. In the absence of any gate controls, all the gates are in open state, during the execution of the list.

In cases where the execution time of the list is greater than the cycletime, the list is truncated and the next iteration starts when the current time equals BTR + CTR.

Table 485. GCL and associated registers - BTR and CTR

Current time	Gate control applied	Accumulator value	BTR (with updates)
14200	OOCCCCCC	1000	14200
15200	OOOOCCCC	1500	14200
15700	CCCCOOCC	3700	14200
17900	CCCCCCOO	4000	14200
18200	OOOOOOOO	0	20200
20200	OOCCCCCC	1000	20200
21200	OOOOCCCC	1500	20200

Table 485 describes an example in which the execution starts at 14200 and the first set of gate controls "OOCCCCCC" are applied immediately. The time interval is 1000 ns, so the next set of gate controls are applied at 14200 (BTR) + 1000 (Accumulator) = 15200 ns . The above table shows that there are no gate controls available after the execution of the last gate control and before the next iteration of the loop. The gates are deemed in open state during that time period as depicted at 18200 current time.

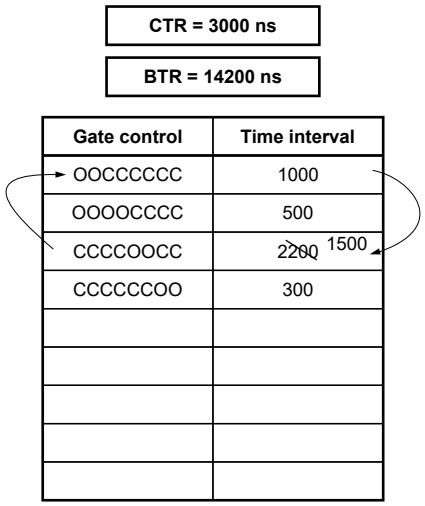


Figure 403. GCL and associated registers - BaseTime and CycleTime list execution time > CycleTime

Table 486 indicates that the list execution takes longer than the allocated CycleTime, so the list is truncated and the list starts from the BTR+CTR.

Table 486. GCL and associated registers - BTR and CTR, execution time > Cycle Time

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
14200	OOCCCCCC	1000	14200
15200	OOOOCCCC	1500	14200
15700	CCCCOOCC	3700	14200
17200	OOCCCCCC	1000	17200
18200	OOOOCCCC	1500	17200
18700	CCCCOOCC	3700	17200

When you apply the third set of gate controls, BTR + CycleTime (17200) < BTR + Accumulator (17900), so the list is truncated and execution switches to a new iteration at 17200.

72.10.11.3.2 Installing a new GCL

The switch to the new GCL can happen in one of the following ways when a new software programmed GCL is available and executed at the new BTR value:

- New base time aligned with current schedule
- New base time unaligned with current schedule

72.10.11.3.2.1 New base time aligned with current schedule

If the choice of cycle time for the new gating cycle is unchanged from the cycle time for the current gating cycle, and if the BTR chosen for the new gating cycle (new BTR) is an integer multiple of the current cycle time (+ current BTR), then the new gating cycle exactly lines up with the old gating cycle, that is, the cycle start times for the new gating cycle is same as they would have been for the old configuration. This could be considered to be the ideal case and allows the new gating cycle to be installed and executed with no timing issues. The implementation completes the execution of an iteration of the current list and switches to the new list at the beginning of the BaseTime listed in the new list.

If (New base time > = Current time)

ConfigChangeTime = New BaseTime

Else If (New base time < Current time)

1. Set the BTRError

2. ConfigChangeTime = (New BaseTime + N* New CycleTime)

where N is the smallest integer for which the relation ConfigChangeTime >= CurrentTime and (N =< 8) is true.

When N > 8 the hardware cannot auto recover and the loop count value in BTRError reporting is set to 1111 requires the software to reprogram the new base time.

Figure 404 illustrates the installation of the new GCL along with the timelines.

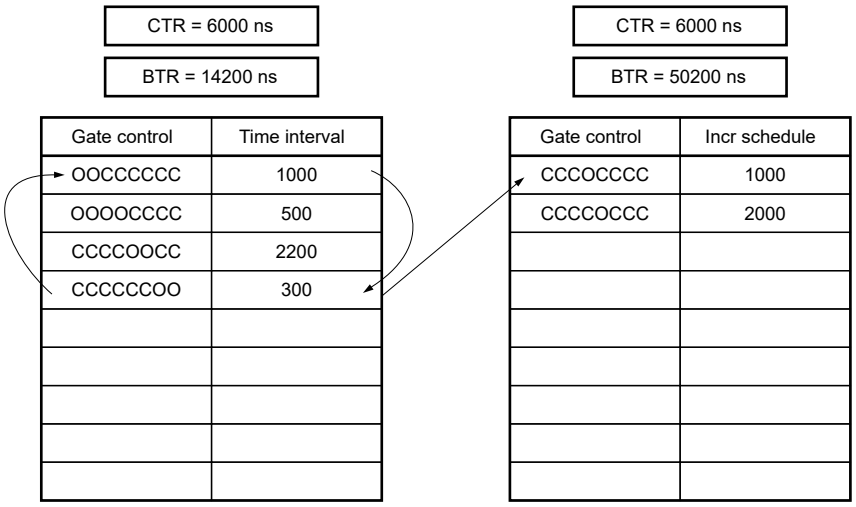


Figure 404. Switching to a new configuration that is aligned with the existing configuration

In the above example, after the sixth iteration of the first GCL, the BTR values of the old and new GCL are equal. At that point the new GCL is processed as a natural extension to the existing GCL.

Table 487. GCL and associated registers - BTR and CTR

Current time	Gate control applied	Accumulator value	BTR (with updates)
44200	OOCCCCCC	1000	44200
45200	OOOOCCCC	1500	44200
45700	CCCCOOCC	3700	44200

Table continues on the next page...

Table 487. GCL and associated registers - BTR and CTR (continued)

Current time	Gate control applied	Accumulator value	BTR (with updates)
47900	CCCCCCOO	4000	44200
48200	OOOOOOOO	0	50200
50200	CCCOCCCC	1000	50200
51200	CCCCOCCC	3000	50200
53200	OOOOOOOO	0	56200

72.10.11.3.2.2 New base time unaligned with current schedule

If the new cycletime differs from the current cycletime or new basetime in the future and is not an integer multiple of current cycletime, then the old and new cycles do not line up. When new basetime is reached (when the new configuration is installed and starts to execute), the last old cycle is normally truncated to start the first new cycle. This could be undesirable if it results in a very short last old cycle; arguably it would be better to simply extend the penultimate old cycle by that small amount, rather than starting a very short cycle. The Cycle Time Extension Register (related to the current config list) allows this extension of the last old cycle to be done in a defined way; if the last complete old cycle ends normally in less than current Cycle Time Extension (TER) ns before the new base time, then the last complete cycle before new BaseTime is reached is extended so that it ends at new BaseTime.

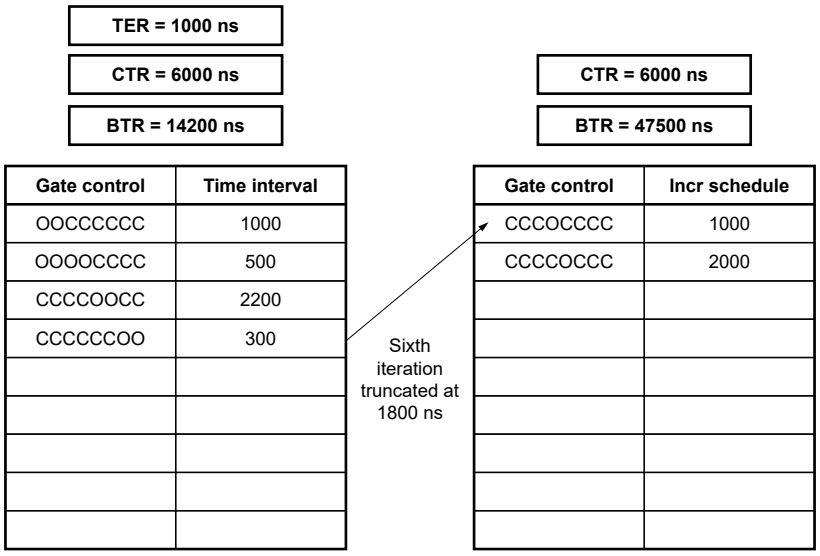


Figure 405. Switching to new list truncating the current list

At the end of the fifth iteration the Current time + Cycle time extension (TER) < New BTR so the sixth iteration of current configuration is started. During the sixth iteration of the current list when the new BTR value is smaller than the next schedule in the current list, it switches to the new list.

Table 488. Extending to new list by truncating the current list

Current time	Gate control applied	Accumulator value	BTR (with updates)
44200	OOCCCCCC	1000	44200

Table continues on the next page...

Table 488. Extending to new list by truncating the current list (continued)

Current time	Gate control applied	Accumulator value	BTR (with updates)
45200	OOOOCCCC	1500	44200
45700	CCCCOOCC	3700	44200
47500	CCCOCCCC	4000	44200
48500	CCCCOCCC	0	50200
50500	OOOOOOOO	1000	50200

Below is an example where the current config list is extended instead of starting a new iteration as the extension time of 800 ns is less than the allowed cycle extension time (TER) of 1000 ns.

Figure 406. Switching to new list by extending the current list

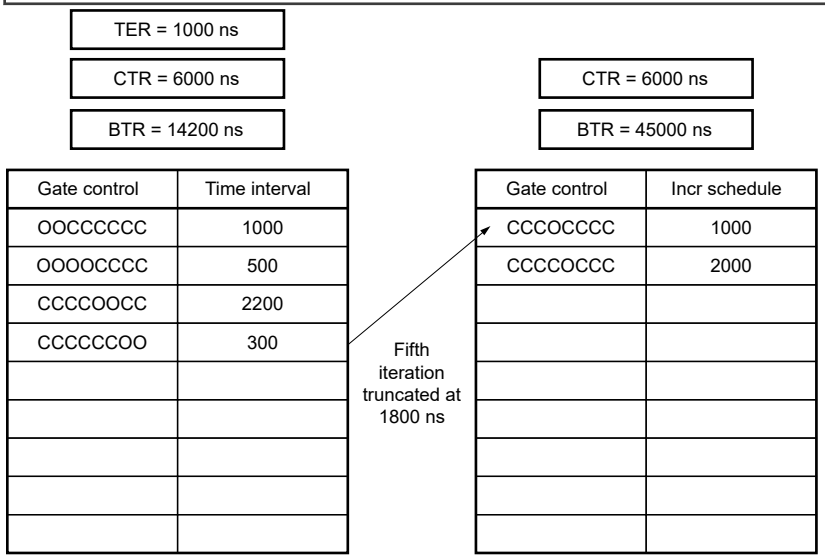


Table 489. Switching to new list by extending the current list

Current time	Gate control applied	Accumulator value	BTR (with updates)
38200	OOCCCCCC	1000	38200
39200	OOOOCCCC	1500	38200
39700	CCCCOOCC	3700	38200
41900	CCCCCCOO	4000	38200
42200	OOOOOOOO	0	44200
45000	CCCOCCCC	1000	45000

Table continues on the next page...

Table 489. Switching to new list by extending the current list (continued)

Current time	Gate control applied	Accumulator value	BTR (with updates)
46000	CCCCOCCC	3000	45000
48000	OOOOOOOO	0	51000

72.10.11.4 Impact of Transmit Scheduling Algorithms on EST

When EST is used in isolation, the Gate Control List manages the final open/close state of the Queues along with the checks carried out by the Transmission Selection Algorithm in MTL. As the Gate Controls operate on a predefined repetitive schedule, it is recommended to use Strict Priority or Credit Based Shaper (CBS) scheduling algorithms.

Other algorithms such as the Weighted Round Robin (WRR), Deficit Weighted Round Robin (DWRR) and Weighted Fair Queuing (WFQ) implement masking of the queues based on the current winning queue. The algorithm is applied only among the group of queues that open simultaneously. To ensure Queues whose gates are "Open" get priority, these algorithms are modified to treat "gate open" queues and "gate closed" queues as separate groups giving priority to gate open queues.

For example, consider 4 queues (Q3, Q2, Q1, Q0) with weights 4:3:2:1; Q3 and Q2 are in Open state in slot one slot, while Q1 and Q0 are in Open state in another slot. In this case, the scheduler works as follows:

1. In the first slot, the Q3 and Q2 are serviced in the ratio of 4:3 for the duration the slot is open.
2. In the second slot, the queues Q1 and Q0 are serviced in the ratio of 2:1.
3. Fresh arbitration is started every time a slot is opened.

In other words, the traffic does not get distributed in the intended ratio of 4:3:2:1; but as two groups with different ratios and only for the duration of the slot when the gates are open continuously

NOTE

See Programming guidelines for EST

72.10.12 Frame preemption (FPE)

Frame preemption breaks the interfering frame into smaller fragments. Therefore, the guard band needs to be only as large as the largest possible interfering fragment instead of the largest possible interfering frame. During the guard band, only the frames that can complete the transmission of the entire frame before the next gate close event are permitted. This ensures that the high priority traffic can always start at the beginning of the window reserved for it.

Preemption allows one or more higher priority (express) frames to interrupt the transmission of a lower priority (preemptable) frame; the preemptable frame transmission is resumed and completed after the express frame transmission is complete. To support frame preemption, the following two abstractions of the MAC are used:

- A preemptable MAC, called pMAC, which carries the preemptable traffic.
- An express MAC, called eMAC, which carries the express traffic.

In the implementation, only parts of the MAC that holds the state during preemption is replicated and represented as pMAC and eMAC. The MAC uses the following two ways to puts on hold, the transmission of the preemtable traffic, in the presence of express traffic:

- The MTL scheduler interrupts the preemptable traffic that is currently being transmitted. When the preemption capability is active, the MAC interrupts the transmission and reception of preemptable frames. A preempted fragment can be followed by zero or more express frames, before the continuation fragments. The preemptable frame can be fragmented any number of times, however, the minimum final and non-final fragment length criterion must be is met. However, interleaving of more than 1 preemptable packet is not permitted. This implies that if a preemptable packet is fragmented by an express packet, another preemptable packet cannot be transferred until all the remaining fragments of the first preempted packet are transferred.

- The MTL scheduler prevents starting the transmission of preemptable traffic. When the preemption capability is inactive, the pMAC entity is disabled and only express traffic is transmitted or received. Frame preemption feature can be enabled by setting EFPE field of the MAC_FPE_CTRL_STS register.

If you are unable to schedule fragmented packets after a certain number of attempts because of some reason, there is a possibility that the remaining fragmented packets are dropped. For this, software can program MTL_EST_Control[LCSE] to increase number of iterations or MTL_EST_Control[DFBS] to disable dropping of packet.

NOTE

EST and FPE cannot be used at the same time.

72.10.12.1 GCL Modification to Support FPE

In the EST-only configuration, the GCL entry has up to 24 bits of Time Interval and up to 8 high order bits representing the Gate Open/Close state requirements as shown in the following figure.

Gate control (up to 8 bits)	Time interval (ns) 16, 20 or 24 bits
OOCCCCCC	T0
OOOOCCCC	T1
CCCCOOCC	T2
CCCCCCOO	T3
|	|
|	|
OCOCOCOO	T126
OOOOCCCC	T127

Figure 407. Gate Control List When FPE is Disabled

EST only supports SetGate operation, which implies that the gates are set to either open or close at a given time interval. However, when both Frame Preemption (FPE) and EST are enabled, the GCL also supports Set-and-Hold-MAC and Set-and-Release-MAC operations, in addition to the SetGate operation. To enable the support of hold and release operations, the format of the GCL is slightly changed while configuring and enabling the FPE. The first Queue (Q0) is always programmed to carry preemption traffic and therefore it is always Open. The bit corresponding to Q0 is renamed as Release/Hold MAC control. The TX Queues whose packets are preemptable are indicated by setting the PEC field of the MTL_FPE_CTRL_STS register. The GCL bit of the corresponding Queue are ignored and considered as always "Open". So, even if the software writes a "0" ("C"), it is ignored for such queues.

Gate control (up to 7 bits)	Release or hold indication	Time interval (ns) 16, 20 or 24 bits
CCCOOOO	0	T0
CCCCOOO	0	T1
OCCCOOO	1	T2
COCCOOO	1	T3
CCOCOOO	1	T4
CCCOOOO	0	T5
CCCCOOO	0	T6

Figure 408. Gate Control List When FPE is Enabled

When the Release/Hold bit transitions from a '0' to '1', a Set-and-Hold-MAC operation is performed. This marks the cease of the preemptable traffic. This is achieved by sending a Hold request to MTL "ha" ns in advance (where ha is the time interval mentioned in the Hold Advance (HADV) field of the MTL_FPE_Advance register). When the Release/Hold bit transitions from a '1' to '0' a Set-and-Release-MAC operation is performed. This marks the resuming of the preemptable traffic. This is achieved by sending a Release request to MTL "ra" ns in advance (where ra is the time interval mentioned in the Release Advance (RADV) field of the MTL_FPE_Advance register). The preemptable traffic is blocked for the time duration the Release/Hold bit is set to a '1' in the Gate Control List.

72.10.12.2 Impact of Preemption on CBS

In Credit Base Shaper(CBS), the definition of "Transmit" is as follows:

- TRUE for the duration of frame transmission from the queue.
- FALSE when frame transmission from the queue is complete.
- When CBS algorithm is used along with frame preemption, the value of "Transmit" is TRUE only while the frame is being transmitted by the MAC. If the frame transmission is delayed or interrupted (for instance, a preemptable frame transmission is interrupted to allow the transmission of an express frame from a different queue, or the start of express frame is delayed because a preemptable frame is being transmitted) the value of "Transmit" is FALSE until transmission of the frame commences or is resumed.

Also, the value of "Transmit" is FALSE during the transmission of any overhead that is a consequence of frame preemption. For example, additional frame overhead (mCRC, Fragment Count) that is added to the preemptable frame.

At any given time, if there are no frames in the queue, and the value of Transmit is FALSE, and credit is positive value, the credit value is set to zero if here is no preemptable frame from the queue for which transmission is in progress but has been interrupted.

72.10.12.3 mPacket Format

When the preemption capability is active, MAC sends mPackets to the PHY. An mPacket can be one of the following:

1. A express packet
2. A preemptable packet
3. An initial fragment of a preemptable packet
4. A continuation fragment of a preemptable packet

Start mPacket Delimiter (SMD)

The value of the SMD indicates whether the mPacket contains an express packet, the start of a preemptable packet (initial fragment or complete packet), or any of continuation fragments of a preemptable packet. Following table shows the valid SMD values.

Table 490. Possible SMD Values of mPacket

mPacket Type	Notation	Frame Count	Value
verify packet	SMD-V	-	0x07
respond packet	SMD-R	-	0x19
express packet	SMD-E	-	0xD5
preemptable packet start	SMD-S0	0	0xE6
	SMD-S1	1	0x4C

Table continues on the next page...

Table 490. Possible SMD Values of mPacket (continued)

mPacket Type	Notation	Frame Count	Value
	SMD-S2	2	0x7F
	SMD-S3	3	0xB3
continuation fragment	SMD-C0	0	0x61
	SMD-C1	1	0x52
	SMD-C2	2	0x9E
	SMD-C3	3	0x2A

frag_count

A frag_count is a modulo-4 counter that increments for each continuation fragment of the preemptable packet. The frag_count protects against mPacket reassembly errors by enabling detection of the loss of up to 3 packet fragments.

The frag_count field is present only in mPackets with SMD-C notation (continuation fragment). The frag_count is zero in the first continuation fragment of each preemptable packet.

Table 491. Possible frag_count Values

frag_count	Value
0	0xE6
1	0x4C
2	0x7F
3	0xB3

mData

The contents of the packet from the MAC, starting with the first byte after the SFD to the last byte before the FCS are sent in the mData fields of one or more mPackets for that frame. The minimum size of the mData field is 60 bytes.

CRC The CRC field contains a cyclic redundancy check (CRC) and has an indication of the final mPacket of a frame. In the final mPacket of a frame, the CRC field contains the last 4 octets of the MAC frame (the FCS field).

For other mPackets, the CRC field contains an mCRC value. The mCRC is calculated on the octets of the packet from the first octet of the frame (the octet following the SFD of preemption frames) to the last octet of the packet transmitted in that mPacket by performing an XOR of the calculated 32 bit CRC value of the fragment and a value of 0x0000FFFF.

Summary of Packet Formats

- Express Packet: 7bytes of PREAMBLE, SMD-E, Data, and CRC
- Complete Preemptable Packet: 7 bytes of PREAMBLE, <Current Preemptable packet SMD>, Data, CRC
- Initial Fragment (non-final) of Preemptable Packet: 7 bytes of PREAMBLE, <Current Preemptable packet SMD>, Data, mCRC
- Continuation fragments (non-final) of Preemptable packet: 6 bytes of PREAMBLE, <Current Preemptable continuation fragment SMD>, <Current Preemptable continuation fragment FC>, Data, mCRC

- Final fragment of Preemptable packet: 6 bytes of PREAMBLE, <Current Preemptable continuation fragment SMD>, <Current Preemptable continuation fragment FC>, Data, CRC

Table 492. Current and Previous SMD Values

Previous Preemptable packet SMD	Current Preemptable packet SMD
SMD-S0	SMD-S1
SMD-S1	SMD-S2
SMD-S2	SMD-S3
SMD-S3	SMD-S0

Table 493. Current and Previous SMD Values

Previous Preemptable fragment SMD	Previous Preemptable fragment FC	Current Preemptable continuation fragment SMD	Current Preemptable continuation fragment FC
SMD-S0	NA	SMD-C0	FC0
SMD-S1	NA	SMD-C1	FC0
SMD-S2	NA	SMD-C2	FC0
SMD-S3	NA	SMD-C3	FC0
SMD-C0	FC0	SMD-C0	FC1
SMD-C0	FC1	SMD-C0	FC2
SMD-C0	FC2	SMD-C0	FC3
SMD-C0	FC3	SMD-C0	FC0
SMD-C1	FC0	SMD-C1	FC1
SMD-C1	FC1	SMD-C1	FC2
SMD-C1	FC2	SMD-C1	FC3
SMD-C1	FC3	SMD-C1	FC0
SMD-C2	FC0	SMD-C2	FC1
SMD-C2	FC1	SMD-C2	FC2
SMD-C2	FC2	SMD-C2	FC3
SMD-C2	FC3	SMD-C2	FC0
SMD-C3	FC0	SMD-C3	FC1

Table continues on the next page...

Table 493. Current and Previous SMD Values (continued)

Previous Preemptable fragment SMD	Previous Preemptable fragment FC	Current Preemptable continuation fragment SMD	Current Preemptable continuation fragment FC
SMD-C3	FC1	SMD-C3	FC2
SMD-C3	FC2	SMD-C3	FC3
SMD-C3	FC3	SMD-C3	FC0

72.10.12.4 Transmit Preemption

When FPE is enabled (setting EFPE=1 in MAC_FPE_CTRL_STS register), the MTL preempts a preemptable frame, when a "hold" request is asserted (EST/Qbv configured and enabled) express frames are available for transmission, that is, frame is present in MTL FIFO and is qualified for arbitration, after ensuring that the minimum mPacket mData field size is met. Therefore, preemption occurs only if at least 60 bytes of the preemptable frame have been transmitted and at least 64 bytes (including the frame CRC) remain to be transmitted.

The earliest starting position of preemption is controlled by the AFSZ field of the MTL_ FPE_CTRL_STS Register. Preemption does not occur until at least 64 * (1+ AFSZ) - 4 bytes of the preemptable frame have been sent.

When preemption occurs, all the preemptable queues are blocked and only the express queues are allowed to arbitrate (if more than one express queue has traffic) and transmit.

Continuation fragment of the preempted frame is the first frame to be transmitted after "release" request is asserted (EST/Qbv configured and enabled) and all the express traffic transmission completes.

NOTE

All the PTP packets should be transmitted as express packets

MTL communicates the following frame-type information to the MAC using a 2-bit preemption control signal on the MTI interface qualified by SoF and EoF.

- Express Frame
- Preemption Frame (Full or Fragment)
- Continuation Fragment (non-Final or Final)

Table 494. Preemption Control Values on MTI Interface for Various Frame Types

Qualifier	Preemption Control Value	Frame Type
SoF	00	Start Express
SoF	01	Start Preemption
SoF	10	Continuation Fragment
SoF	11	Reserved
EoF	00	End of Frame
EoF	01	End of Fragment

MTL should wait for the previous fragment status to be received before resuming the continuation fragments of a preempted frame. Fragment status is described in detail in the Tx Fragment Status section.

72.10.12.4.1 MAC Tx Preemption

MAC supports preemption by implementing the functionality needed to generate the mPackets as described in “mPacket Format”.

Based on the Preemption Control value received on the MTI interface (qualified with SoF and EoF), MAC determines the frame type (shown in the above table 'Preemption Control Values on MTI Interface for Various Frame Types') and generates mPackets accordingly.

When the preemption capability is active, MAC replaces the SFD of a preemption packet with an SMD-S value. A 2-bit rolling frame count is encoded in the SMD-S value.

The SMD-E value is the same as the SFD value so the SFD of an express packet does not need to be replaced.

72.10.12.4.1.1 Tx Fragment Status

MAC sends Tx fragment status to indicate successful transmission of fragmented mPackets.

In case of a transmission error (underflow, jabber, and so on) the frame status is sent (and not a fragment status) with an error indication along with all other relevant status fields. In case of receiving an error status for a transmitted fragment, the MTL drops the remaining fragments and does not send any more continuation fragments.

72.10.12.5 Receive Preemption

72.10.12.5.1 MAC Receive Preemption

When FPE is enabled, the MAC Receiver pases the incoming packets and differentiates between Express packets and preemptable packets. An SMD containing an SMD-E indicates express packet, and SMD containing an SMD-S indicates the first mPacket of a preemptable packet.

If an mPacket containing an SMD-S is received when MAC has not completed receiving the previous preempted packet, MAC sets a CRC Error status for the previously received partial packet.

When an SMD-S is detected, MAC records the frame count indicated by the SMD and then begins sending data on the MRI interface.

The MAC checks the last four bytes of the mPacket . If the last four bytes of the mPacket do not match CRC, that indicates the end of the packet with or without a CRC error as per the CRC check result. If the last four octets of the mPacket match, that indicates that the packet was preempted.

An SMD containing an SMD-C indicates an mPacket that continues the data for a preempted packet. Upon receiving an SMD value of SMD-C, MAC checks the following:

1. A preempted packet is in progress
2. The frame count indicated by the SMD matches the frame count of the packet in progress
3. The frag_count value indicates the next fragment count.

If any of the above check fails, the mPacket is discarded and MAC sets a CRC Error status for the partially received packet.

If all the checks pass, the next fragment count is incremented modulo 4.

When a packet is preempted, the MAC saves the state of the partially received packet (filter check status, timestamp, length fields etc.) and will be able to process any received Express packets before the continuation fragment is received.

The MAC Receiver sends a "dummy status" for all the mPacket fragments successfully received and sends the Rx status with the final fragment. If an error is detected during any of the fragments the Rx status is sent and the fragment is marked as final fragment. All subsequent continuation fragments received for this packet are dropped in the MAC.

The MAC communicates the following frame type information to the MTL using a 2-bit preemption control signal.

1. Express Frame
2. Preemption Frame (Full or Fragment)
3. Continuation Fragment (non-Final or Final)

Table 495. Preemption Control Values on MRI Interface for Various Frame Types

Qualifier	Preemption Control Value	Frame Type
SoF	00	Start Express
SoF	01	Start Preemption
SoF	10	Continuation Fragment
SoF	11	Reserved
EoF	00	End of Frame
EoF	01	End of Fragment

72.10.12.5.1.1 Data Alignment

When a received frame cannot be fragmented on any byte boundary, MAC retains the unaligned bytes of data in the previous fragment and resends them with the next fragment as shown in the following figure.

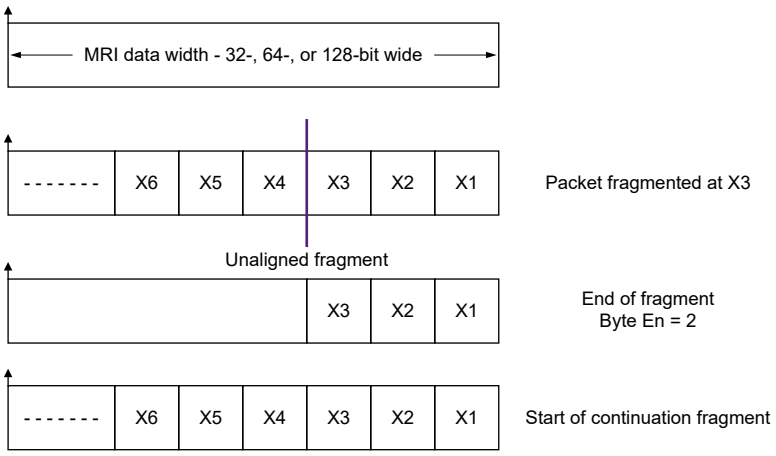


Figure 409. Data Alignment Feature of MAC

72.10.12.5.1.2 MTL Receive Preemption

The MTL Rx must have at least 2 Rx queues to support FPE function as the preemptable packets and the express packets must be routed to separate Rx queues. The destination Rx queue of a received packet is controlled by MAC_RxQ_Ctrl[n] registers. Program these registers such that Preemptable traffic and express traffic are not routed to the same Rx queue. As the queue mapping for tagged packets is based on VLAN user-priority field, this implies that priority of preemptable and express packets are mutually exclusive. In other words, packets of a certain priority (traffic class) are either express or preemptable but cannot be both.

For the preemptable frames, in addition to the PSRQ (Priority Selected in Rx queue) based queue routing, a programmable "Frame Preemption Residue Queue" (FPRQ) is supported to route all other Preemption Packets received (Untagged, SA/ DA or VLAN Filter Fails forwarded due to RA being set or VTFE being reset).

Table 496. Preemption Control Values on MRI Interface for Various Frame Types

Preemption Packet Type	Queue Routing
AV Tagged Packets passing the SA/DA and VLAN filters	PSRQ*
DCB/Generic Tagged Packets passing the SA/DA and VLAN filters	PSRQ**
Tagged Packets failing the SA/DA and VLAN filters without setting the Receive All (RA = 0) or setting VTFE = 1	Dropped
All other packets OR when RA = 1	FPRQ

72.10.12.5.1.3 MTL Receive Arbitration

On the ARI Interface, data is fetched from the MTL Rx FIFO based on the arbitration selected in the MTL_Operation_Mode register. Frame based arbitration can be used only when all the MTL Preemption Queues are operating in Store and Forward mode, else there will be loss of bandwidth on the ARI interface because all the fragments of a preemptable packet is not available. Therefore, any express packets received between the fragments are blocked until all the fragments are received and transferred; this defeats the purpose of express traffic.

When operating in either Threshold (cut through) or Store and forward modes of operation, PBL based arbitration is recommended over Frame based arbitration. In case of PBL based arbitration, the watermark check is always performed and the arbitration/ transfer of data in terms of chunks of "PBL" size of data which is almost similar to the concepts of "fragments". Therefore the express queue packets get blocked for less time as well as the ARI interface transfers the data without loss of efficiency.

72.10.12.6 Verify and Respond mPackets

When FPE function is present, the MAC can receive and detect the Verify and Respond mPackets, even when FPE is not enabled by software. When MAC detects valid Verify/Respond mPackets, it notifies the software by setting the RVER and RRSP fields of MAC_FPE_CTRL_STS register respectively. Optionally an interrupt can be generated. As such packets have empty (all-zero) data payload, they are dropped inside the MAC and not forwarded to the MRI.

Software can set the SVER and SRSP fields of MAC_FPE_CTRL_STS register to request MAC to transmit Verify and Respond mPackets respectively. Upon successful transmission of these frames, MAC clears the SVER/SRSP bits and sets the TVER & TRSP fields of MAC_FPE_CTRL_STS register. Optionally an interrupt can be generated when these events occur.

Software must ensure that frame preemption verify and response packet are not triggered at the same time. Trigger either of the packets and then wait for its completion before triggering another packet.

72.10.12.7 Frame Preemption and MMC Counter and Interrupt Registers

The following MMC counters and associated Interrupt registers are instantiated/present in the MAC when Frame Preemption feature is enabled.

Table 497. MMC Counters and Associated Interrupt Registers

Frame Assembly Error Counter	Description	Associated MMC Counter
Frame Assembly Error Counter	A 32-bit counter that provides the number of MAC frames with reassembly errors on the Receiver, due to mismatch in the Fragment Count value.	MMC_Rx_Packet_Assembly_Err_Cntr

Table continues on the next page...

Table 497. MMC Counters and Associated Interrupt Registers (continued)

Frame Assembly Error Counter	Description	Associated MMC Counter
Frame SMD Error Counter	A 32-bit counter that provides the number of received MAC frames rejected due to arriving with an SMD-C when there was no preceding preempted frame.	MMC_Rx_Packet_SMD_Err_Cntr
Frame Assembly OK Counter	A 32-bit counter that provides the number of MAC frames that were successfully reassembled.	MMC_Rx_Packet_Assembly_Ok_Cntr
MAC Rx Fragment Counter	A 32-bit counter that provides the number of additional mPackets received due to preemption.	MMC_Rx_FPE_Fragment_Cntr
MAC Tx Fragment Counter	A 32-bit counter that provides the number of additional mPackets transmitted due to preemption.	MMC_Tx_FPE_Fragment_Cntr
Hold Request Counter	A 32-bit counter that maintains the count of number of times a hold request is given to MAC.	MMC_Tx_Hold_Req_Cntr

72.10.12.7.1 Additional Registers Associated With MMC Interrupts

Following are the additional registers associated with MMC interrupts for the MMC error counters:

- MMC_FPE_Tx_Interrupt
- MMC_FPE_Tx_Interrupt_Mask
- MMC_Rx_Interrupt
- MMC_FPE_Rx_Interrupt_Mask

72.10.13 Time-Based Scheduling

Time-based scheduling feature is suitable for traffic whose periodicity and rate are predictable. To improve the quality-of-service of such traffic,

- The transmit DMA fetches the packet from the host memory for transmission at designated time. This helps the software to setup the Transmit descriptors in advance even before packet is ready/available. It reduces the overhead on the software and avoids constant monitoring of the time and preparing descriptors just in time when the packet is targeted to be transmitted.
- The MAC transmits the packet only at the designated/pre-determined time even if the packets are fetched in advance. This helps in maintaining a constant transmission rate that can be consumed by the receiver station; therefore avoiding congestion and excessive buffering in the network.

The time-based scheduling feature supports fetching and launching an Ethernet packet at (or after) a pre-determined time. The time-based scheduling is supported only in the following modes/configurations:

- Full duplex mode
- Link speed is 100Mbps or higher

The following figure shows the time-based scheduling flow.

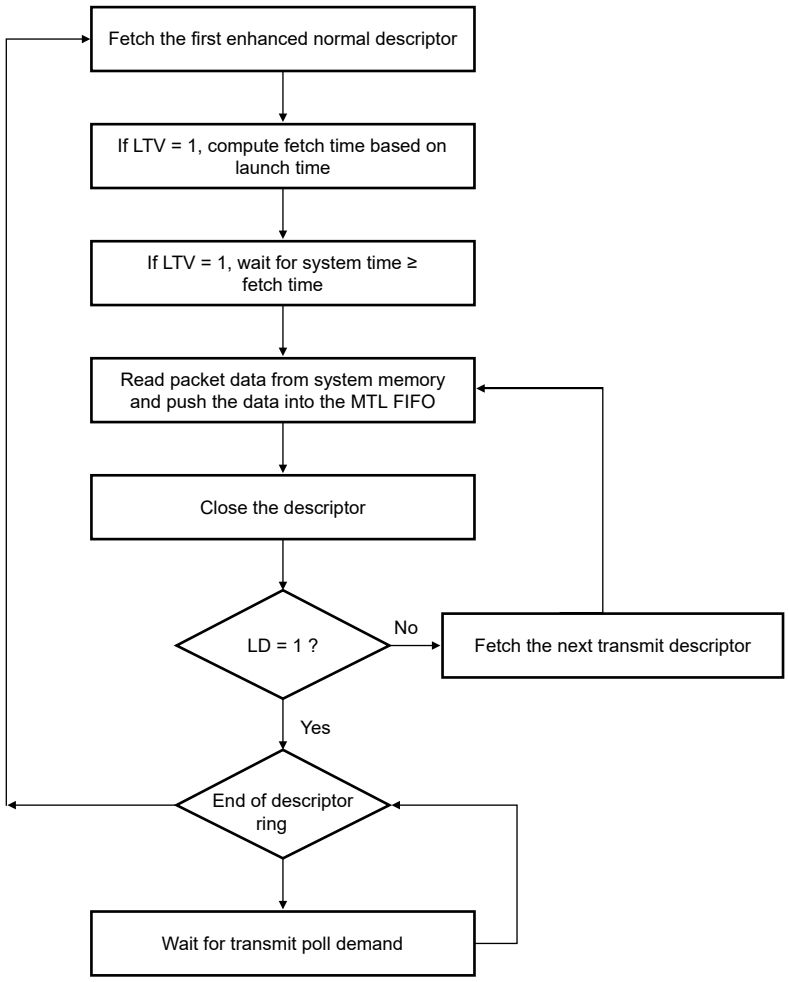


Figure 410. Time-Based Scheduling Flow

Following are the definitions specific to the time-based scheduling feature:

72.10.13.1 Launch time

The time beyond which MTL can schedule the packet for transmission. The launch time resets after the transmission of one frame. These are the two formats of the launch time.

Normal or absolute format:

In this format, the launch time is interpreted in the normal or absolute format if the ESTM bit of the MTL_TBS_CTRL register is set to 0.

The launch time is a 32-bit value, where most-significant 8-bits represent the time in seconds and the rest 24-bits represent the time in 256 ns. The launch time is compared against the IEEE 1588 based system or PTP time (bits[39:8]) and rolls over after 256 seconds.

The maximum value of the lower 32-bits of system time is 999,999,999 decimal (0x3B9AC9FF) and it wraps to 0 when reaching this value (representing a full second). Therefore, the maximum value of the lower 24-bits of the launch time (after multiplying by 256) must be 0x3B9AC9.

As the maximum value of launch time is 256 seconds,

- Launch time is greater than current system time when its value is between system time[39:8] and System time[39:8] + 128 sec.

- Launch time is less than current system time when its value is between System Time[39:8] + 128sec and System Time[39:8] + 256 sec, because this is a modulo 256 computation.

EST or offset format

In this format, the launch time is a offset value relative to the time which the BTR of the GCL list provided in the GCL slot number (GSN) indicates. The value in the BTR is always updated to the start time of the current loop of the GCL.

For each packet, the GSN value and the launch time value are specified in the enhanced transmit descriptor in the DMA configurations, or as control word in the MTL configurations.

The launch time offset is a 32-bit value; the upper 8-bits represent the time in seconds and the rest 24 bits represent the time in 256 ns, which is added to the BTR value corresponding to GCL slot number. The value of the launch time offset must be smaller than the value of the cycle time (specified in the CTR register that is implemented when EST is enabled). If the CTR is greater than or equal to 1s, the maximum value of the lower-24 bits of launch time offset must be 999,999,999 decimal (0x3B9AC9FF).

In this format, the launch time is a 64-bit value, which is interpreted as, Launch time = Launch GSN BTR[63:0] + (Launch time offset[31:0] << 8), which is compared with the system time [63:0].

GSN BTR is the Base Time value at which the Launch GSN loop started execution.

GCL slot number (GSN) A modulo 16 count of the GCL loop count is implemented and it is known as GSN. The count is incremented for every new GCL loop. Installation of new GCL list does not impact the count. Read MTL_EST_Status[CGSN] to obtain the current GCL slot number.

The maximum value of GSN is 15. Therefore the GSN values between MTL_EST_Status[CGSN] and MTL_EST_Status[CGSN]+8 represent current or future slots and all other GSN values are interpreted as elapsed slots or past slots. So, for the correct interpretation of time, GSN value must be between MTL_EST_Status[CGSN] and MTL_EST_Status[CGSN]+8.

72.10.13.2 Launch expiry time

Normal or Absolute Mode

In this mode, when MTL_TBS_CTRL[LEOV] = 1, MTL_TBS_CTRL[LEOS] determines the maximum amount of time a frame is eligible for launch, starting from the time the frame becomes eligible for launch.

The launch expiry offset is a 24-bit value defined in 256 ns units, with a maximum possible value of 999,999,999 ns (0x3B9AC9FF).

Launch expiry time = (Launch time[39:8] + LEOS[32:8]) * 256 ns

The packet with a specific launch time is considered as eligible for transmission when the launch time is less than the system time and (if MTL_TBS_CTRL[LEOV] = 1) the system time is less than the launch expiry time.

When the system time is greater than the launch expiry time, the frame is categorized as expired and it drops from the MTL FIFO.

NOTE

- For correct interpretation and meaningful operation, the fetch, launch, and launch expiry time must never set to a value larger than the Current system time + 128 sec; such a value is interpreted as time that has already elapsed.
- In Full Duplex mode, the frames dropped from the MTL FIFO have error summary (Bit 15) and excessive deferral (bit 3) of TxStatus set.

EST/Offset Mode

In EST mode, when MTL_TBS_CTRL[LEOV] = 1, MTL_TBS_CTRL[LEGOS] and MTL_TBS_CTRL[LEOS] determines the maximum amount of time a frame remains eligible for launch, starting from the time the frame becomes eligible for launch.

Launch Expiry Offset = (LEGOS: LEOS)

LEGOS holds the GSN offset (multiples of CTR time) and LEOS holds maximum value of CTR (sub CTR values) in ns.

Launch expiry offset is a 24-bit value defined in the units of 256 ns, with a maximum possible value of the smaller of 999,999,999 ns or CTR-1 ns.

The Launch expiry GSN is computed as follows:

Launch expiry GSN = (Launch GSN + LEGOS + CCMA) where, CCMA is the CTR carry due to modulo addition. This value is 1 if ((Launch time offset + LEOS) << 8) is equal to or greater than CTR.

When CCMA = 1, Launch expiry offset = (Launch time offset+ LEOS) - CTR

When CCMA = 0,

Launch expiry offset = (Launch time offset + LEOS).

Launch expiry time = Launch expiry GSN BTR [63:0] + Launch expiry time offset.

When MTL_TBS_CTRL[LEOV] is not 1, the launch expiry time is not checked.

When MTL_TBS_CTRL[LEOV] = 1, and

- the system time is greater than the launch expiry time, the frame drops from MTL FIFO. Then the frame is considered as expired.
- the launch time is smaller than the system time and the launch expiry time is greater than the system time, then the frame is considered eligible for launch.

NOTE

- Max value of MTL_TBS_CTRL[LEGOS] is 7. This indicates that when MTL_TBS_CTRL[LEOV] = 1, the frame has a maximum life time of <8 GCL loop iterations after it becomes eligible for launch.
- The slot number of the first GCL list that executes each time after EST is enabled, is zero.

72.10.13.3 Fetch time

This accounts for all possible delays in the DMA fetch operation and ensures that the frame is present in the MTL FIFO before the launch time.

If DMA_TBS_CTRL[FTOV] is 1, it indicates the fetch time for each packet. If DMA_TBS_CTRL[FTOV] is not 1, it indicates that the fetch time offset is not valid and DMA fetches packets without any time constraints.

Normal/Absolute mode

In this mode fetch time is derived or calculated by reducing the time specified in DMA_TBS_CTRL[FTOS] from the given launch time.

In Normal mode, the fetch launch time is computed as:

Fetch Time[39:8] = (Launch Time[39:8] - FTOS[31:8])

The fetch time is 32-bits and is compared against the system time[39:8] to determine whether it is eligible for fetching the frame:

- The fetch time is defined as greater than system time if the fetch time is in the range of system time[39:8]" and system time[39:8] + 128 sec. The frame is considered as not-eligible for fetch.
- The fetch time is defined as smaller than the system time if the fetch time is in the range of system time[39:8] + 128 sec and system time[39:8] + 256 sec. The frame is considered as eligible for fetch.

This is a modulo 256 computation.

EST/Offset mode

In this mode, the fetch GSN offset (DMA_TBS_CTRL[FGOS]) provides the slot number offset to deduct from the launch GSN. In this case the FTOS value must be smaller than 999,999,999 ns or CTR-1 ns.

If (Launch time offset >= FTOS):

Fetch time offset = ((Launch time offset - FTOS) * 256ns)

CBFS(CTR borrow for fetch subtraction) = 0

If (Launch time offset < FTOS) Fetch time offset = CTR + ((Launch time offset - FTOS) * 256ns)

CBFS (CTR borrow for fetch subtraction) = 1

The fetch GSN is computed as follows:

Fetch GSN = Launch GSN - FGOS - CBFS

Fetch time = Fetch GSN BTR[63:0] + Fetch time offset

The frame is eligible for DMA fetch when the fetch time is smaller than the system time.

DMA operations sequence

This is the sequence of operation when FTOV = 1:

1. Fetch the first enhanced normal descriptor (FD = 1).
2. Compute the fetch time on the basis of the launch time and wait for the system time to be greater than fetch time, if LTV = 1 and fetch is enabled.
3. Read the frame (data) from the host memory and transfer to MTL FIFO.
4. Close the normal descriptor.
5. Fetch the next normal descriptor (if the previous descriptor was not the last).
6. Repeat steps 4 to 6, until the last descriptor of the frame (LD = 1).

After the last descriptor of a frame, program the next enhanced normal descriptor with a new launch time and with LTV = 1. Otherwise, process the subsequent frames without any time restrictions.

See Programming the launch time in time-based scheduling for more information.

72.11 TCP/IP Offloading Features

This section and all other sections, along with respective sub-sections are Synopsys Proprietary. Used with permission.

72.11.1 Transmit Checksum Offload Engine

The MAC has a Checksum Offload Engine (COE) to support checksum calculation and insertion in the Transmit path, using which, the software can offload the task of checksum insertion to the hardware. In the transmit path MAC calculates the checksum and inserts it in the Tx packet. This feature helps in reducing the load on the software and can improve the overall throughput of the system. This feature is supported for only Q0 queue.

The checksum offload engine module supports two types of checksum calculation and insertion. The checksum engine can be controlled for each packet by setting the CIC bits (TDES3 Bits[17:16]).

Note: The checksum for TCP, UDP, or ICMP is calculated over a complete packet, and then inserted into its corresponding header field. Because of this requirement, when this function is enabled, the Tx FIFO automatically operates in the store-and-forward mode even if IP is configured for Threshold (cut-through) mode.

72.11.1.1 IP Header Checksum Engine

In IPv4 datagrams, the integrity of the header fields is indicated by the 16-bit Header Checksum field (the eleventh and twelfth bytes of the IPv4 datagram). The COE detects an IPv4 datagram when the Type field of Ethernet packet has the value 0x0800 and the Version field of IP datagram has the value 0x4. The checksum field of the input packet is ignored during calculation and replaced with the calculated value.

NOTE

IPv6 headers do not have a checksum field. Therefore, the COE does not modify the IPv6 header fields.

The result of this IP header checksum calculation is indicated by the IP Header Error status bit in the Transmit status (Bit 0 in Table 19-12 on page 1330). This status bit is set whenever the values of the Ethernet Type field and the Version field of IP header are not consistent, or when the Ethernet packet does not have enough data, as indicated by the IP header Length field. In other words, this bit is set when an IP header error is asserted under the following circumstances:

- For IPv4 datagrams:
  - The received Ethernet type is 0x0800, but the Version field of IP header is not equal to 0x4.
  - The IPv4 Header Length field indicates a value less than 0x5 (20 bytes).
  - The total packet length is less than the value given in the IPv4 Header Length field.
- For IPv6 datagrams:
  - The Ethernet type is 0x86dd but the IP header Version field is not equal to 0x6.
  - The packet ends before the IPv6 header (40 bytes) or extension header (as given in the corre-sponding Header Length field in an extension header) is completely received.

72.11.1.2 TCP/UDP/ICMP Checksum Engine

The TCP/UDP/ICMP Checksum Engine processes the IPv4 or IPv6 header (including extension headers) and determines whether the encapsulated payload is TCP, UDP, or ICMP. The checksum is calculated for the TCP, UDP, or ICMP payload and inserted into its corresponding field in the header. The Tx COE can work in the following two modes:

- The TCP, UDP, or ICMPv6 pseudo-header is not included in the checksum calculation and is assumed to be present in the Checksum field of the input packet. This engine includes the Checksum field in the checksum calculation, and then replaces the Checksum field with the final calculated checksum.
- The engine ignores the Checksum field, includes the TCP, UDP, or ICMPv6 pseudo-header data into the checksum calculation, and overwrites the checksum field with the final calculated value.

NOTE

For ICMP-over-IPv4 packets, the Checksum field in the ICMP packet must always be 16'h0000 in both modes, because pseudo-headers are not defined for such packets. If it does not equal 16'h0000, an incorrect checksum may be inserted into the packet.

The result of this operation is indicated by the Payload Checksum Error status bit in the Transmit Status vector (Bit 12 in Table 19-3 on page 1324). This engine sets the Payload Checksum Error status bit when it detects that the packet has been forwarded to the MAC Transmitter engine in the store-and-forward mode without the end of packet (EOP) being written to the FIFO, or when the packet ends before the number of bytes indicated by the Payload Length field in the IP Header is received. When the packet is longer than the indicated payload length, the COE ignores them as stuff bytes, and no error is reported. When this engine detects the first type of error, it does not modify the TCP, UDP, or ICMP header. For the second error type, it still inserts the calculated checksum into the corresponding header field. Following table describes the functions supported by Transmit Checksum Offload engine based on the packet type. When the MAC does not insert the checksum, it is indicated as "No" in the table.

NOTE

You should not enable checksum insertion for IPv4 or IPv6 packets that are greater than the frame size constraint specified in Description of Transmit Checksum Offload Engine because it may result in incorrect checksum insertion or unexpected behavior.

Table 498. Transmit Checksum Offload Engine Functions for Different Packet Types

Packet Type	Hardware IP headerchecksum insertion	Hardware TCP/ UDPchecksum insertion
Non-IPv4 or IPv6 packet	No	No
IPv4 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in MAC) but less than or equal to the frame size constraint specified in Transmit Checksum Offload Engine.	Yes	Yes

Table continues on the next page...

Table 498. Transmit Checksum Offload Engine Functions for Different Packet Types (continued)

Packet Type	Hardware IP headerchecksum insertion	Hardware TCP/ UDPchecksum insertion
IPv6 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in MAC) but less than or equal to the frame size constraint specified in Transmit Checksum Offload Engine.	Not Applicable	Yes
IPv4 with TCP, UDP, or ICMP	Yes	Yes
IPv4 packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet is an IPv4 fragment	Yes	No
IPv6 packet with the following next header fields in main or extension headers: • Hop-by-hop options (in IPv6 main header) • Hop-by-hop options (in IPv6 extension header) • Destinations options • Routing (with segment left 0) • Routing (with segment left > 0) • TCP, UDP, or ICMP • Authentication • Any other next header field in main or extension headers	• Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable	• Yes • No • Yes • No • No • Yes • No • No
IPv4 packet has TCP header with Options fields	Yes	Yes
IPv4 Tunnels: • IPv4 packet in an IPv4 tunnel • IPv6 packet in an IPv4 tunnel	• Yes (IPv4 tunnel header) • Yes (IPv4 tunnel header)	• No • No
IPv6 Tunnels: • IPv4 packet in an IPv6 tunnel • IPv6 packet in an IPv6 tunnel	• Not applicable • Not applicable	• No • No
IPv4 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Yes	Yes
IPv6 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Not applicable	Yes
IPv4 frames with security features (such as encapsulated security payload)	Yes	No
IPv6 frames with security features (such as encapsulated security payload)	Not applicable	No

72.11.2 Receive Checksum Offload Engine

IP provides the Checksum Offload Engine that is used to detect any error in an IPv4 or IPv6 packet in the receive path. The MAC verifies the checksum field of the received packet with the internally calculated checksum and provides the status.

The Receive Checksum Offload engine is used to detect errors in IP packets by calculating the header checksum and further matching it with the received header checksum. This engine also identifies a TCP, UDP, or ICMP payload in received IP packets and calculates the checksum of such payloads appropriately.

Here, both IPv4 and IPv6 packet in the received Ethernet packets are detected and processed for data integrity. The MAC receiver identifies IPv4 or IPv6 packets by checking for value 0x0800 or 0x86DD, respectively, in the Type field of the received Ethernet packet. This identification is applicable to single VLAN-tagged packets. It is also applicable to double VLAN-tagged packets when the Enable Double VLAN Processing option is selected and the EDVLP bit of the MAC_VLAN_Tag register is set.

The Rx COE calculates the IPv4 header checksums and checks that they match the received IPv4 header checksums. The result of this operation (pass or fail) is given to the RFC module for insertion into the receive status word. The IP Header Error bit is set for any mismatch between the indicated payload type (Ethernet Type field) and the IP header version, or when the received packet does not have enough bytes, as indicated by the Length field of the IPv4 header (or when fewer than 20 bytes are available in an IPv4 or IPv6 header).

This engine also identifies a TCP, UDP, or ICMP payload in the received IP datagrams (IPv4 or IPv6) and calculates the checksum of such payloads properly, as defined in the TCP, UDP, or ICMP specifications. This engine includes the TCP, UDP, or ICMPv6 pseudo-header bytes for checksum calculation and checks whether the received checksum field matches the calculated value. The result of this operation is given as a Payload Checksum Error bit in the receive status word. This status bit is also set if the length of the TCP, UDP, or ICMP payload does not match the expected payload length given in the IP header.

Following table describes the functions supported by the Rx COE based on the packet type. When the payload of an IP packet is not processed (indicated as "No" in the table), the information (whether the checksum engine is bypassed or not) is given in the receive status.

NOTE

The MAC does not append any payload checksum bytes to the received Ethernet packets.

Table 499. Receive Checksum Offload Engine Functions for Different Packet Types

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
Non-IPv4 or IPv6	No	No
IPv4 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in the MAC)	Yes	Yes
IPv6 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in the MAC)	Not Applicable	Yes
IPv4 with TCP, UDP, or ICMP	Yes	Yes
IPv4 header's protocol field contains a protocol other than TCP, UDP, or ICMP	Yes	No

Table continues on the next page...

Table 499. Receive Checksum Offload Engine Functions for Different Packet Types (continued)

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
IPv4 packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet is an IPv4 fragment	Yes	No
IPv6 packet with the following next header fields in main or extension headers: • Hop-by-hop options (in IPv6 main header) • Hop-by-hop options (in IPv6 extension header) • Destinations options • Routing (with segment left 0) • Routing (with segment left > 0) • TCP, UDP, or ICMP • Any other next header field in main or extension headers	• Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable	• Yes • No • Yes • Yes • No • Yes • No
IPv4 packet has TCP header with Options fields	Yes	Yes
IPv4 Tunnels: • IPv4 packet in an IPv4 tunnel • IPv6 packet in an IPv4 tunnel	• Yes (IPv4 tunnel header) • Yes (IPv4 tunnel header)	• No • No
IPv4 Tunnels: • IPv4 packet in an IPv6 tunnel • IPv6 packet in an IPv6 tunnel	• Not Applicable • Not Applicable	• No • No
IPv4 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled)	Yes	Yes
IPv6 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled)	Not Applicable	Yes
IPv4 frames with security features (such as encapsulated security payload)	Yes	No

Table continues on the next page...

Table 499. Receive Checksum Offload Engine Functions for Different Packet Types (continued)

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
IPv6 frames with security features (such as encapsulated security payload)	Not Applicable	No

Set the IPC bit of the MAC_Configuration register for enabling Receive Checksum offload.

72.12 MAC Management Counters

This section is Synopsys Proprietary. Used with permission.

IP supports storing the statistics about the received and transmitted packets in registers that are accessible through the application/software. The MMC module maintains a set of registers for gathering statistics on the received and transmitted packets. The MMC counters are free running.

For MMC register details, see registers "MMC_Control" to "MMC_Rx_FPE_Fragment_Cntr" in EMAC register descriptions.

72.13 Flow Control

This section and its sub-sections are Synopsys Proprietary. Used with permission.

72.13.1 Transmit Flow Control

The Transmit Flow Control involves transmitting Pause packets in full-duplex mode and backpressure in half-duplex mode to control the flow of packets from the remote end.

72.13.1.1 Flow control in Full duplex mode

IP supports the IEEE802.3x Pause Packets.

Pause packet structure is shown in the following table.

Table 500. Pause Packet Fields

Field	Description
DA	Contains the special multicast address
SA	Contains the MAC address 0
Type	Contains 8808
MAC Control opcode	Contains 0001 for IEEE 802.3x Pause Control packets; 0101 for PFC packets
PT	Contains Pause time specified in the PT field of the MAC_Q#_Tx_Flow_Ctrl register

72.13.1.2 Pause Packet Control

When the FCB bit is set, the MAC generates and transmits a single Pause packet. If the FCB bit is set again after the Pause packet transmission is complete, the MAC sends another Pause packet irrespective of whether the pause time is complete or not. To extend the pause or terminate the pause prior to the time specified in the previously-transmitted Pause packet, the application should program the Pause Time register with appropriate value and then again set the FCB bit.

72.13.1.3 Flow Control in Half-Duplex Mode

In half-duplex mode, the MAC uses the deferral mechanism for the flow control (backpressure). When the application requests to stop receiving packets, the MAC sends a JAM pattern of 32 bytes when it senses a packet reception, provided the transmit flow control is enabled. This results in a collision and the remote station backs off. If the application requests a packet to be transmitted, it is scheduled and transmitted even when the backpressure is activated. If the backpressure is kept activated for a long time (and more than 16 consecutive collision events occur), the remote stations abort the transmission because of excessive collisions.

Following table describes the flow control in the Tx path for Queue 0 based on the setting of the following bits:

- EHFC bit of MTL_RxQ0_Operation_Mode register
- TFE bit of MAC_Q0_Tx_Flow_Ctrl register
- DM bit of MAC_Configuration register

Flow control is similar for all queues.

Table 501. Tx MAC Flow Control

EHFC	TFE	DM	Description
x	0	x	The MAC transmitter does not perform the flow control or backpressure operation.
0	1	0	The MAC transmitter performs back-pressure when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1.
1	1	0	The MAC transmitter performs back-pressure when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1. In addition, the MAC Tx performs back-pressure when Rx Queue level crosses the threshold set by Bits[10:8] of MTL_RxQ0_Operation_Mode register.
0	1	1	The MAC transmitter sends the Pause packet when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1.
1	1	1	The MAC transmitter sends the Pause packet when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1. In addition, the MAC Tx sends a Pause packet when Rx Queue level crosses the threshold set by Bits[10:8] of MTL_RxQ0_Operation_Mode register.

72.13.1.4 Enabling Transmit Flow Control

To independently enable Transmit Flow Control for each Tx queue, set the TFE bit in the MAC_Q#_Tx_Flow_Ctrl register.

72.13.2 Receive Flow Control

In the Receive path, the Flow Control is functional only in the full-duplex mode. If any Pause packet is received in the half-duplex mode, the packet is considered as a normal control packet.

NOTE

Receive pause packets should have a frame size of 64 bytes.

The Receive Flow Control is implemented by the MAC based on the bit value of the respective register, and the destination address and different fields of the received packet.

Following table describes the flow control in the Rx path based on the setting of the following bits:

- RFE bit of MAC_Rx_Flow_Ctrl register
- DM bit of MAC_Configuration register

Table 502. Rx MAC Flow Control

RFE	DM	Description
0	x	The MAC receiver does not detect the received Pause packets.
1	0	The MAC receiver does not detect the received Pause packets but recognizes such packets as Control packets.
1	1	The MAC receiver detects or processes the Pause packets and responds to such packets by stopping the MAC transmitter.

72.13.2.1 Enabling Receive Flow Control

To enable Pause Flow Control, set the RFE bit in the MAC_Rx_Flow_Ctrl register.

72.14 Loopback Mode

This section is Synopsys Proprietary. Used with permission.

The MAC supports Loopback of transmitted packets to its receiver.

The following are some guidelines for using the loopback mode:

- Enable loopback only with the full-duplex mode. In half-duplex mode, the carrier sense signal (crs) or collision (col) signal inputs get sampled which may result into issues such as packet dropping.
- If the loopback mode is enabled without connecting a PHY chip (for example, in FPGA setup), you should externally generate the Tx and Rx clocks and provide these clocks to the MAC.
- Do not loop back big packets. Big packets (>1522 bytes) may get corrupted in the loopback FIFO.

To enable this feature, program the LM bit of the MAC_Configuration register.

You can enable loopback for all PHY interfaces. The data is always looped back through internal asynchronous FIFO on to the internal Receive MII or GMII interface, irrespective of which PHY interface is selected. The loopback data is also passed through the corresponding transmit PHY interface block, on to the Ethernet line.

72.15 Automotive Safety Features

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

IP supports the automotive safety feature.

72.15.1 Error Correction Code (ECC) Protection for Memories

The Error Correction Code (ECC) block can correct single-bit error and detect double-bit error.

At the write interface, ECC checkbits are generated by computing ECC on the contents of the data bus and respective address is appended with the data that is written to the memory.

At the read interface, ECC checkbits are recomputed on the content of the read data and the respective address is compared with the received checkbits in the memory.

Following figure shows the block diagram of ECC protection for memories

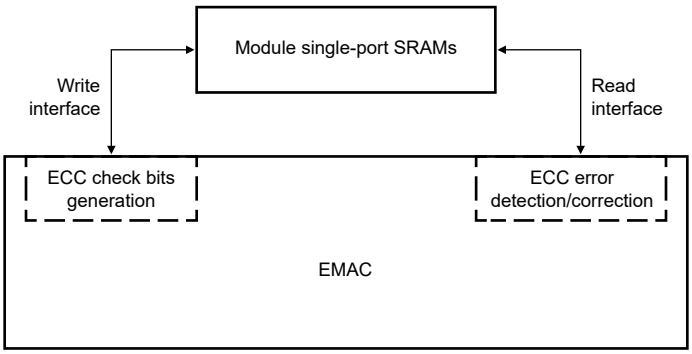


Figure 411. Block Diagram of ECC Protection for Memories

72.15.1.1 Handling the Address Mismatch

The ECC is calculated over the sum of memory data and memory location address. However only data is written in the memory along with the ECC(address is excluded).

On the read interface, the ECC is checked over the sum of memory DATA, memory ECC, and Internally generated address. When an ECC correctable error is detected over the range of the address bit position, it is treated as an uncorrectable error. This is because data might have been read from a different location.

72.15.1.2 Diagnostic Support for the Error Management

Following statistics are provided for monitoring the error behavior on each of the memory blocks.

- Error status provided to management
  - A separate status for correctable, uncorrectable, and address mismatch is specified in the MTL_EC-C_Interrupt_Status register.
  - Memory locations at which correctable and uncorrectable errors are detected is specified in the MTL_ECC_Err_Addr_Status register. In addition, a control bit (MEEAO field of MTL_ECC_Control register) decides, whether the first errored memory address is reported or the latest errored memory address is reported.
  - MTL_ECC_Err_Cntr_Status register has separate counters to count the number of correctable and uncorrectable errors
- Interrupts provided to management
  - Separate interrupts are generated for correctable and uncorrectable errors.
  - sbd_sfty_ue_intr_o interrupt is generated when uncorrectable errors are detected and these errors cannot be masked.
  - sbd_sfty_ce_intr_o interrupt is generated when correctable errors are detected. The module sets the respective interrupt enable bits (in the DMA/MTL_ECC_Interrupt_enable register).
  - To find the root cause of the error, read the DMA/MTL_Safety_Interrupt_Status and DMA/MTL_ECC_Interrupt_Status registers.
  - To clear the interrupt, write 1 in the respective interrupt status bit in DMA/MTL_ECC_Inter-rupt_Status registers.

NOTE

- The status/counters/interrupts are generated per memory.
- MTL Tx and Rx memory is logically divided into multiple queues. But, ECC diagnostics (status/counter/interrupts) are common to all queues.

72.15.1.3 ECC Error Injection Capabilities

IP supports error injection capabilities for each memory as a static configuration. The position where the errors are injected in the data word is random. For each memory, 3 control bits are provided to inject errors. The 3 bits are:

- A single bit to enable error injection
- Two bits to indicate the type of error to be injected ❑ 00: 1 bit error
  - 01: 2 bit error
  - 10: 3 bit error
  - 11: 1 bit error in address field

The control bit descriptions for

- MTL Tx/Rx and DMA TSO memory are specified in the MTL_DBG_CTL register, for and for R.
- MTL EST memory are specified in the MTL_EST_GCL_Control register
- Rx parser memory are specified in the MTL_RXP_Indirect_Acc_Control_Status register

NOTE

While using debug mode for ECC error injection:

- There should be no traffic in the module
- When multiple CSR writes are required for writing single data word into the memory, the application should ensure that all the CSR writes corresponding to one memory write main-tains the same value for the error injection control word.
- See Programming guidelines for ECC protection for memories

72.15.2 Finite State Machine(FSM) Parity and Timeout Protection

The FSM protection feature supports FSM parity and time out protection.

72.15.2.1 FSM State Parity Protection

Odd parity is implemented on all the FSM state register bits. Parity is monitored for every clock, after the reset is de-asserted. When a bit flips due to transient errors or permanent faults, the erroneous FSM is set to its default state and an uncorrectable error is indicated.

When the FSM state parity protection is enabled, by setting PRTYEN field of MAC_FSM_Control Register, the FSM state error is indicated by setting the FSMPES field of the MAC_DPP_FSM_Interrupt_Status register. Also, the Safety Interrupt (sbd_sfty_ue_intr_o) is asserted when an FSM Error is detected.

Error Injection mode is also supported for FSM parity error check. Program the Error Injection enable for the respective clock domain denoted by [23:16] bits of MAC_FSM_Control Register.

72.15.2.2 FSM Timeout Protection

The FSM Timeout feature provides a mechanism to ensure that all FSMs in the module can complete a transaction and return to a known completion state (IDLE or any other state that indicates completion of a transaction/transfer) within the programmed timeout value.

Program the TMR field of MAC_FSM_ACT_Timer register with a value that indicates the number of CSR cycles required to generate a 1 microsecond tic. This microsecond tic is internally used to generate the programmed timeout duration. Two timeout tics (normal mode timeout and large mode timeouts) are generated to provide flexibility to choose one per clock domain, by using the bits [31:24] of MAC_FSM_Control Register. Supported values for Timeout duration are 1us, 1.024ms, 16.384ms, 65.5536ms, 262.144ms, 1048.576ms (~1sec), 4194.304ms (~4s), 8.388s, 16.777s, and 33.554s which is based on the programmable bits NTMRMD, LTMRMD fields of MAC_FSM_ACT_Timer register. Interface timeouts are based on the normal mode tic generation; only the FSM timeouts depend on either normal or large tick selection.

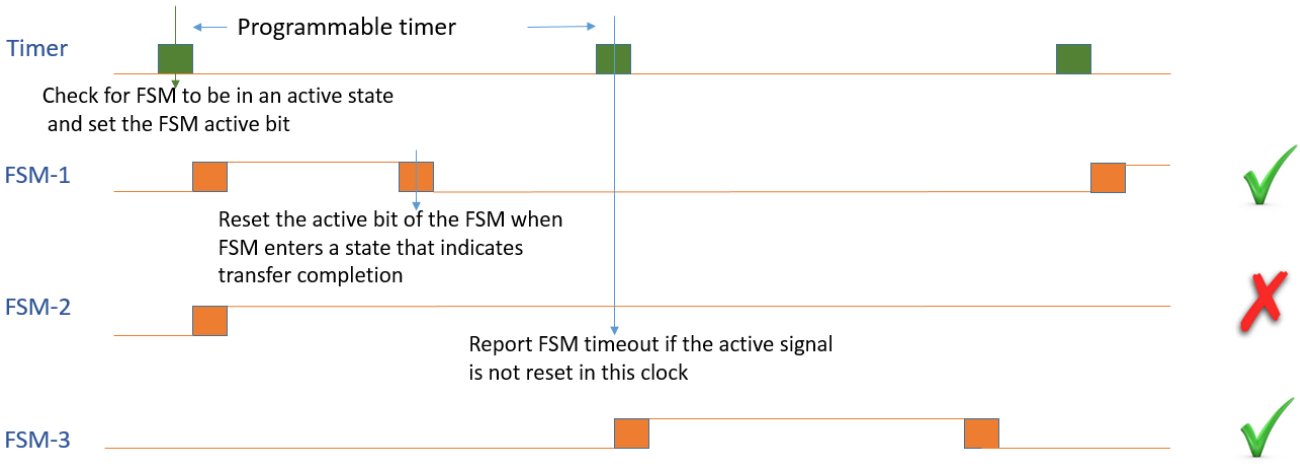


Figure 412. FSM Timeout Timing Diagram

In the above figure, the timer ticks are generated based on the programmable timeout value. The timer ticks are synchronized and made available in all the clock domains that have the FSMs.

At every timer tick, all the FSMs are monitored for being in active state (non-IDLE state, which indicates the FSM is actively processing transactions or hand-shakes) and an FSM active flag bit is set. The active flag bit is implemented for every FSM that is monitored for timeout. When the FSM reaches an IDLE or transaction completion state, the active flag bit of that FSM is reset. At the subsequent timer tick, all the FSMs' active flag bits are monitored and any flag that is set indicates a timeout for that FSM. This process of setting the flag and checking at subsequent timer tick is repeated at every tick.

Timeout Error Injection per clock domain is enabled by programming [15:8] fields of MAC_FSM_Control register. When Timeout error injection enable is set for a clock domain, the FSMs in that clock domain automatically timeout and generate an interrupt even without traffic. As error injection is a debug mode, TMR value need not indicate 1us, but can be programmed to a smaller value at which debug mode ticks are needed. FSM Timeout and Parity Error Injection modes can be used for testing at key-on/key-off.

The FSM timeout status is specified in the [15:8] field of MAC_DPP_FSM_Interrupt_Status register as per the respective clock domains. One FSM timeout status bit is made available per clock domain. The safety interrupt (sbd_sfty_ue_intr_o) is asserted when the timeout error is set for any of the clock domains. IP does not attempt to recover from a FSM timeout condition and relies on the application to take the corrective action (resetting IP).

72.15.2.3 Enabling FSM Parity and Timeout Protection

FSM timeout feature is enabled by setting the TMOUTEN field of MAC_FSM_Control register.

NOTE

See Programming guidelines for FSM parity and timeout

72.15.3 Application/CSR Interface Timeout Protection

This feature provides timeout protection to the application/CSR Interface. All the interfaces which has the handshake mechanism monitored for potential hangs due to the external agent (Master/Slave/Interconnect/Application client) not responding to the requests/transfers initiated by the QoS. After the request is initiated the response arrival interval is monitored. If the response does not arrive within a programmed time (TMR field of MAC_FSM_ACT_Timer) the timeout is triggered, using similar trigger generation as explained in the Section FSM protection.

The timeout is triggered when IP initiates SEQ, NON-SEQ and BUSY transfers on HTRANS[1:0]; and HREADY is not asserted by Slave within the programmed time.

The Timeout status for the AHB master interface is set in the MSTTES field of the MAC_DPP_FSM_Interrupt_Status register. The Safety interrupt (sbd_sfty_ue_intr_o) is asserted when application timeout status is set.

The hardware does not attempt to recover from Application/CSR Interface hangs and relies on software to take appropriate corrective action.

NOTE

- Protection is not needed for APB3 Slave interfaces because potential hangs can be only when IP does not respond to the requests. In such cases the IP internal protection logic (FSM timeout) detects such defects.
- Based on the time at which the transfer is initiated relative to the timer ticks, the timeout period could be any value between the programmed timeout and 2x times the programmed timeout.
- When an uncorrectable safety interrupt is issued and the read of the Interrupt status register returns all zeros, it implies that the CSR read is not functional. As soft reset of IP is not possible without the CSR access, and therefore, a hard reset of IP is recommended.
- See Programming guidelines for FSM parity and timeout .

72.16 Descriptors

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

72.16.1 Overview

The DMA in the Ethernet subsystem transfers data based on a linked list of descriptors. The application creates the descriptors in the system memory. The module supports the following two types of descriptors:

- Normal Descriptor: Normal descriptors are used for packet data and to provide control information applicable to the packets to be transmitted.
- Context Descriptor: Context descriptors are used to provide control information applicable to the packet to be transmitted.

Each normal descriptor contains two buffers and two address pointers. These buffers enable the adapter port to be compatible with various types of memory management schemes.

NOTE

There is no limit for the number of descriptors that can be used for a single packet.

72.16.2 Descriptor Structure

The module supports the ring structure for DMA descriptor as shown in the following figure.

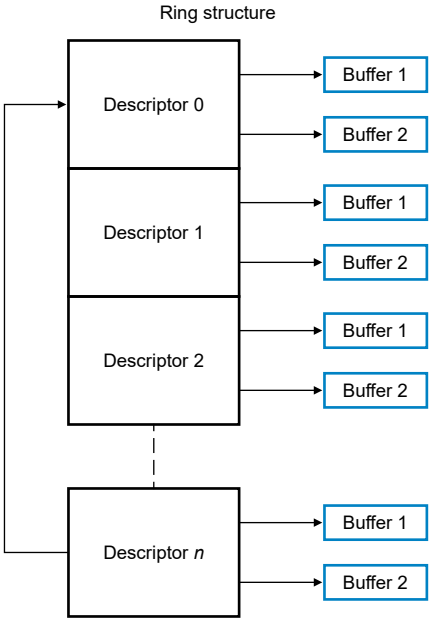


Figure 413. Descriptor Ring Structure

In Ring structure, descriptors are separated by the Word, DWord, or LWord number programmed in the DSL field of the DMA_CH#_Control register. The application needs to program the total ring length, that is, the total number of descriptors in ring span in the following registers of a DMA channel:

- Transmit Descriptor Ring Length Register (DMA_CH#_TxDesc_Ring_Length)
- Receive Descriptor Ring Length Register (DMA_CH#_RxDesc_Ring_Length)

The Descriptor Tail Pointer Register contains the pointer to the descriptor address (N). The base address and the current descriptor pointer decide the address of the current descriptor that the DMA can process. The descriptors up to one location less than the one indicated by the descriptor tail pointer (N – 1) are owned by the DMA. The DMA continues to process the descriptors until the following condition occurs:

Current Descriptor Pointer == Descriptor Tail Pointer;

The DMA goes into the Suspend mode when this condition occurs. The application must perform a write to the Descriptor Tail pointer register and update the tail pointer so that the following condition is true:

Current Descriptor Pointer < Descriptor Tail Pointer;

The DMA automatically wraps around the base address when the end of ring is reached, as shown in the following figure.

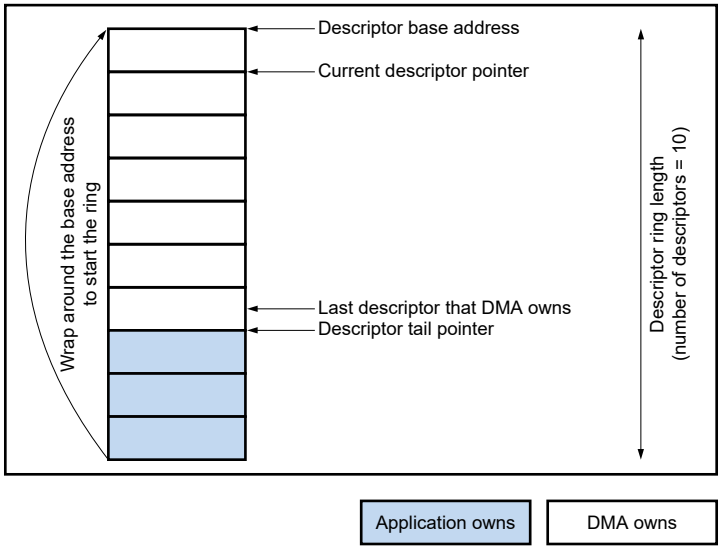


Figure 414. DMA Descriptor Ring

For descriptors owned by the application, the OWN bit of DES3 is reset to 0. For descriptors owned by the DMA, the OWN bit is set to 1. If the application has only one descriptor in the beginning, the application sets the last descriptor address (tail pointer) to Descriptor Base Address + 1. The DMA processes the first descriptor and then waits for the application to advance the tail pointer.

72.16.3 Transmit Descriptor

The DMA in the module requires at least one descriptor for a transmit packet. In addition to two buffers, two byte-count buffers, and two address pointers, the transmit descriptor has control fields which can be used to control the MAC operation on per-transmit packet basis. The Transmit Normal descriptor has two formats: Read format and Write-Back format

NOTE

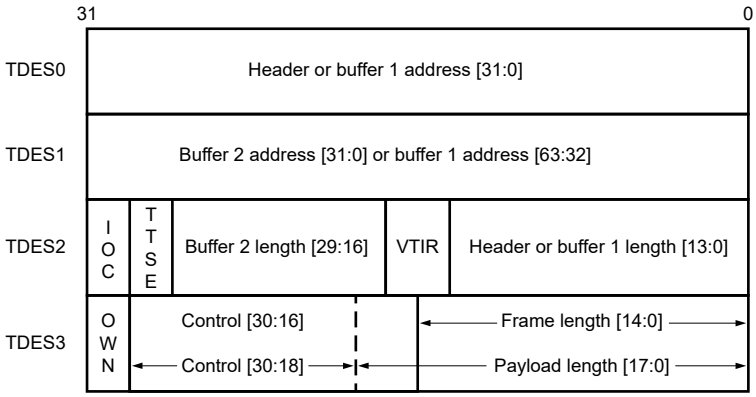
TSO Split header is not supported.

Software must not write same tail pointer addresses when DMA is in Suspended state before setting new descriptor(s) for further processing.

72.16.3.1 Transmit Normal Descriptor (Read Format)

Following figure shows the Read Format for a Transmit normal descriptor. Table: 'TDES0 Normal Descriptor (Read Format)' through Table: 'TDES3 Normal Descriptor (Read Format)' describe the read format for the Transmit Normal Descriptors: TDES0, TDES1, TDES2, and TDES3.

Figure 415. Transmit Descriptor Read Format



72.16.3.1.1 TDES0 Normal Descriptor (Read Format)

Table 503. TDES0 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF1AP	Buffer 1 Address Pointer or TSO Header Address Pointer These bits indicate the physical address of Buffer 1. These bits indicate the TSO Header Address pointer when the following bits are set: • TSE bit of TDES3 • FD bit of TDES3

72.16.3.1.2 TDES1 Normal Descriptor (Read Format)

Table 504. TDES1 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF2AP	Buffer 2 or Buffer 1 Address Pointer This bit indicates the physical address of Buffer 2 when a descriptor ring structure is used. There is no limitation for the buffer address alignment. In 40- or 48-bit addressing mode, these bits indicate the most-significant 8- or 16- bits of the Buffer 1 Address Pointer.

72.16.3.1.3 TDES2 Normal Descriptor (Read Format)

Table 505. TDES2 Normal Descriptor (Read Format)

31	30	29-16	15:14	13:0
IOC	TTSE/ TMWD	B2L	VTIR	HL or B1L

Table 506. TDES2 Normal Descriptor (Read Format)

Bits	Name	Description
31	IOC	Interrupt on Completion This bit controls the setting of TI and ETI status bits in the DMA_CH#_Status register. When ETIC = 1 and TDES2[LD] = 0, this bit sets the ETI bit. When TDES3[LD] = 1, this bit sets the TI status bit.
30	TTSE/ TMWD	Transmit Timestamp Enable or External TSO Memory Write Enable This bit enables the IEEE1588 time stamping for Transmit packet referenced by the descriptor, if TSE bit is not set. If TSE bit is set and external TSO memory is enabled, setting this bit disables external TSO memory writing for this packet.
29:16	B2L	Buffer 2 Length The driver sets this field. When set, this field indicates Buffer 2 length.
15:14	VTIR	VLAN Tag Insertion or Replacement These bits request the MAC to perform VLAN tagging or untagging before transmitting the packets. The application must set the CRC Pad Control bits appropriately when VLAN Tag Insertion, Replacement, or Deletion is enabled for the packet. The following list describes the values of these bits: • 2'b00: Do not add a VLAN tag. • 2'b01: Remove the VLAN tag from the packets before transmission. This option should be used only with the VLAN packets. • 2'b10: Insert a VLAN tag with the tag value programmed in the MAC_VLAN_Incl register or context descriptor. • 2'b11: Replace the VLAN tag in packets with the tag value programmed in the MAC_VLAN_Incl register or context descriptor. This option should be used only with the VLAN packets. These bits are valid when the Enable SA and VLAN Insertion on Tx option is selected while configuring the core.
13:0	HL or B1L	Header Length or Buffer 1 Length For Header length only bits [9:0] are taken. The size 13:0 is applicable only when interpreting buffer 1 length. If the TCP Segmentation Offload feature is enabled through the TSE bit of TDES3, this field is equal to the header length. When the TSE bit is set in TDES3, the header length includes the length in bytes from Ethernet Source address till the end of the TCP header. The maximum header length supported for TSO feature is 1023 bytes. The maximum header length supported for TSO feature is 1023 bytes. If the TCP Segmentation Offload feature is not enabled, this field is equal to Buffer 1 length.

72.16.3.1.4 TDES3 Normal Descriptor (Read Format)

Table 507. TDES2 Normal Descriptor (Read Format)

31	30	29	28	27:26	25:23	22:19	18	17:16	15	14:0
OWN	CTXT	FD	LD	CPC	SAIC	SLOTNUM or THL	TSE	CIC/TPL	TPL	FL/TPL

Table 508. TDES3 Normal Descriptor (Read Format)

Bits	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffer(s).
30	CTXT	Context Type This bit should be set to 1'b0 for normal descriptor.
29	FD	First Descriptor When this bit is set, it indicates that the buffer contains the first segment of a packet.
28	LD	Last Descriptor When this bit is set, it indicates that the buffer contains the last segment of the packet. When this bit is set, the B1L or B2L field should have a non-zero value.
27:26	CPC	CRC Pad Control This field controls the CRC and Pad Insertion for Tx packet. This field is valid only when the first descriptor bit (TDES3[29]) is set. The following list describes the values of Bits[27:26]: • 2'b00: CRC and Pad Insertion The MAC appends the cyclic redundancy check (CRC) at the end of the transmitted packet of length greater than or equal to 60 bytes. The MAC automatically appends padding and CRC to a packet with length less than 60 bytes. • 2'b01: CRC Insertion (Disable Pad Insertion) The MAC appends the CRC at the end of the transmitted packet but it does not append padding. The application should ensure that the padding bytes are present in the packet being transferred from the Transmit Buffer, that is, the packet being transferred from the Transmit Buffer is of length greater than or equal to 60 bytes. • 2'b10: Disable CRC Insertion The MAC does not append the CRC at the end of the transmitted packet. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. • 2'b11: CRC Replacement The MAC replaces the last four bytes of the transmitted packet with recalculated CRC bytes. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. This field is valid only for the first descriptor. Note: When the TSE bit is set, the MAC ignores this field because the CRC and pad insertion is always done for segmentation.
25:23	SAIC	SA Insertion Control These bits request the MAC to add or replace the Source Address field in the Ethernet packet with the value given in the MAC Address 0 register. The application must set the CRC Pad Control bits appropriately when SA Insertion Control is enabled for the packet.

Table continues on the next page...

Table 508. TDES3 Normal Descriptor (Read Format) (continued)

Bits	Name	Description
		Bit 25 specifies the MAC Address Register (1 or 0) value that is used for Source Address insertion or replacement. The following list describes the values of Bits[24:23]: • 2'b00: Do not include the source address • 2'b01: Include or insert the source address. For reliable transmission, the application must provide frames without source addresses. • 2'b10: Replace the source address. For reliable transmission, the application must provide frames with source addresses. • 2'b11: Reserved These bits are valid in the EQOS-DMA, EQOS-AXI, and EQOS-AHB configurations when the Enable SA and VLAN Insertion on Tx option is selected while configuring the core and when the First Segment control bit (TDES3 [29]) is set. This field is valid only for the first descriptor.
22:19	SLOTNUM or THL	SLOTNUM: Slot Number Control Bits in AV Mode These bits indicate the slot interval in which the data should be fetched from the corresponding buffers addressed by TDES0 or TDES1. When the Transmit descriptor is fetched, the DMA compares the slot number value in this field with the slot interval maintained in the RSN field DMA_CH#_Slot_Function_Control_Status. It fetches the data from the buffers only if a value matches. These bits are valid only for the AV channels. THL: TCP/UDP Header Length If the TSE bit is set, this field contains the length of the TCP/UDP header. The minimum value of this field must be 5 for TCP header. The value must be equal to 2 for UDP header. This field is valid only for the first descriptor.
18	TSE	TSO Split header is not supported. The value of this bit is always zero.
17:16	CIC/TPL	Checksum Insertion Control or TCP Payload Length These bits control the checksum calculation and insertion. The following list describes the bit encoding: • 2'b00: Checksum Insertion Disabled. • 2'b01: Only IP header checksum calculation and insertion are enabled. • 2'b10: IP header checksum and payload checksum calculation and insertion are enabled, but pseudo-header checksum is not calculated in hardware. • 2'b11: IP Header checksum and payload checksum calculation and insertion are enabled, and pseudo-header checksum is calculated in hardware. This field is valid when the Enable Transmit TCP/IP Checksum Offload option is selected and the TSE bit is reset. When the TSE bit is set, this field contains the upper bits [17:16] of the TCP Payload (or IP Payload for UDP fragmentation). This allows the TCP/UDP packet length field to be spanned across TDES3[17:0] to provide 256 KB packet length support.

Table continues on the next page...

Table 508. TDES3 Normal Descriptor (Read Format) (continued)

Bits	Name	Description
		This field is valid only for the first descriptor.
15	TPL	Reserved or TCP Payload Length When the TSE bit is reset, this bit is reserved. When the TSE bit is set, this is Bit 15 of the TCP payload length [17:0]. This field is valid only when the Enable TCP Segmentation Offloading for TCP/IP Packets option is selected while configuring the core.
14:0	FL/TPL	Frame Length or TCP Payload Length This field is equal to the length of the packet to be transmitted in bytes. When the TSE bit is not set, this field is equal to the total length of the packet to be transmitted: Ethernet Header Length + TCP /IP Header Length – Preamble Length – SFD Length + Ethernet Payload Length When the TSE bit is set, this field is equal to the lower 15 bits of the TCP payload length in case of segmentation and IP payload in case of UDP fragmentation. In case of segmentation, this length does not include Ethernet header or TCP/UDP/IP header length. In case of fragmentation, this length does not include Ethernet header and IP header. When DWRR/WFQ algorithm is NOT enabled, value written into this field is not used when TSE = 0.

72.16.3.1.5 Transmit normal descriptor (write-back format)

The transmit descriptor write-back format includes timestamp low, timestamp high, OWN, and Status bits.

The write-back format is applicable only for the last descriptor of the corresponding packet. Write 1 to the LD bit (TDES3[28]) in the descriptor where DMA writes back the status and timestamp information for the corresponding transmit packet.

Figure 416 shows the transmit descriptor write-back format.

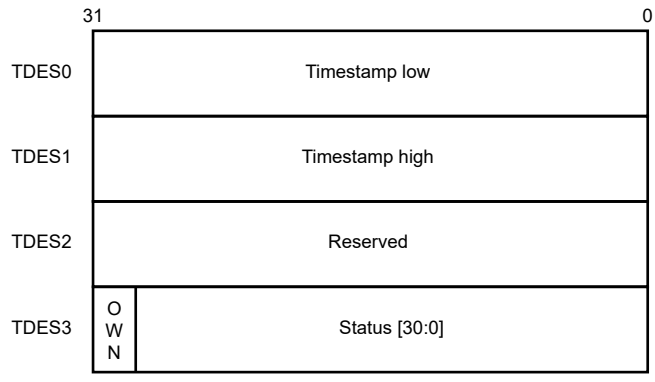


Figure 416. Transmit descriptor write-back format

TDES0 normal descriptor (write-back format)

Table 509 shows that this format is only applicable to the last descriptor of a packet.

Table 509. TDES0 normal descriptor (write-back format)TDES0 normal descriptor (write-back format)

Bit	Name	Description
31:0	TTSL	Transmit Packet Timestamp Low DMA updates this field with least significant 32 bits of the timestamp captured for the corresponding transmit packet. DMA writes the timestamp only if TTSE bit of TDES2 is 1 in the first descriptor of the packet. This field has the timestamp only if you write 1 to the Last Segment bit (LS) in the descriptor and the Timestamp status (TTSS) bit.

TDES1 normal descriptor (write-back format)

Table 510 shows that this format is only applicable to the last descriptor of a packet.

Table 510. TDES1 normal descriptor (write-back format)

Bit	Name	Description
31:0	TTSH	Transmit Packet Timestamp High DMA updates this field with the most significant 32 bits of the timestamp captured for a corresponding transmit packet. DMA writes the timestamp only if the TTSE bit of TDES2 is 1 in the first descriptor of the packet. This field has the timestamp only if you write 1 to the Last Segment bit (LS) in the descriptor and Timestamp status (TTSS) bit.

TDES2 Normal Descriptor (Write-Back Format)

This format is applicable only to the last descriptor of a packet.

Table 511. TDES2 normal descriptor (write-back format)

Bit	Description
31:0	Reserved

TDES3 Normal Descriptor (Write-Back Format)

This format is applicable only to the last descriptor of a packet.

Table 512. TDES3 normal descriptor (write-back format)

31	30	29	28	27:18	17	16	15	14	13	12	11	10	9	8	7:4	3	2	1	0
OWN	CTXT	FD	LD	Rsvd	TTSS	EUE	ES	JT	FF	PCE	LoC	NC	LC	EC	CC	ED	UF	DB	IHE

Table 513. TDES3 normal descriptor (write-back format)

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that the module DMA owns the descriptor. DMA write 0 to this field when it completes the packet transmission. After the write-back is complete, this field is set to 'b0.
30	CTXT	Context Type

Table continues on the next page...

Table 513. TDES3 normal descriptor (write-back format) (continued)

Bit	Name	Description
		This field must be set to 'b0 for normal descriptor.
29	FD	First Descriptor This field indicates that the buffer contains the first segment of a packet.
28	LD	Last Descriptor This field is set to 'b1 for last descriptor of a packet. DMA writes the status fields only in the last descriptor of the packet.
27:24	Rsvd	Reserved
23	DE	Descriptor Error When this field is 1, it indicates that the descriptor content is incorrect. DMA writes 1 to this field during write-back when the descriptor is closed. Descriptor errors can be: • Incorrect sequence from the context descriptor. For example, a location after the first descriptor for a packet. • All 1s • CTXT is 1 along with LD or FD bits as 1. **NOTE** When a descriptor error occurs due to all ones or CTXT, LD, and FD bits are 1, the transmit DMA closes the transmit descriptor with DE and LD bits set to 1. When you write 1 to IOC bit in TDES2 of corresponding first descriptor, the transmit DMA write 1 to TI bit in the DMA_CH#_Status register **NOTE** Based on CTXT, LD, and FD bits of the transmit descriptor, the subsequent descriptor might consider as the first descriptor (even if FD bit is not 1) and partial packet is sent.
22:18	Rsvd	Reserved
17	TTSS	Tx Timestamp Status This field indicates that a timestamp has been captured for the corresponding transmit packet. When this field is 1, it indicates that TDES0 and TDES1 have timestamp values that were captured for the transmit packet. This field is valid only when the Last Segment control bit (TDES3 [28]) in a descriptor is 1. This field is valid only when IEEE1588 timestamping feature is enabled; otherwise, it is reserved.
16	EUE	ECC Uncorrectable Error Status Indicates the ECC uncorrectable error in the TSO memory. **NOTE** An uncorrectable error in the transmit FIFO memory is reported with (Bit 13) FF = 1. This is because the module flushes all such packets.

Table continues on the next page...

Table 513. TDES3 normal descriptor (write-back format) (continued)

Bit	Name	Description
15	ES	Error Summary This field indicates the logical OR of the following bits: • TDES3[0]: IP header error • TDES3[14]: Jabber timeout • TDES3[13]: Packet flush • TDES3[12]: Payload checksum error • TDES3[11]: Loss of carrier • TDES3[10]: No carrier • TDES3[9]: Late collision • TDES3[8]: Excessive collision • TDES3[3]: Excessive deferral • TDES3[2]: Underflow error This field is 1 when EUE (bit 16) = 1.
14	JT	Jabber Timeout This field indicates that the MAC transmitter experiences a jabber time-out. This field is 1 only when MAC_Configuration[JD] is not 1.
13	FF	Packet Flushed This field indicates that DMA or MTL flushes the packet because the CPU gives a software flush command.
12	PCE	Payload Checksum Error This field indicates that the checksum offload engine had a failure and does not insert any checksum into the encapsulated TCP, UDP, or ICMP payload. This failure can either because of insufficient bytes, as the payload length field of the IP header indicates or the MTL starting to forward the packet to the MAC transmitter in Store-and-forward mode without calculating the checksum. This second error condition only occurs when the transmit FIFO depth is less than the length of the Ethernet packet that is transmitted to avoid deadlock. The MTL starts forwarding the packet when the FIFO is full, even in the Store-and-forward mode. This error can also occur when you detects a bus error during packet transfer. When the full checksum offload engine is not enabled, this bit is reserved.
11	LoC	Loss of Carrier This field indicates that a loss of carrier occurred during packet transmission (that is, the gmii_crs_i signal was inactive for one or more transmit clock periods during packet transmission). This field is valid only for the packets transmitted without collision and when MAC operates in the Half-duplex mode.
10	NC	No Carrier This field indicates that the carrier sense signal from the PHY was not asserted during transmission.

Table continues on the next page...

Table 513. TDES3 normal descriptor (write-back format) (continued)

Bit	Name	Description
9	LC	Late Collision This bit indicates that packet transmission was aborted because a collision occurred after the collision window (64 byte times including Preamble in MII mode and 512 byte times including Preamble and Carrier Extension in GMII mode). This bit is not valid if Underflow Error is set.
8	EC	Excessive Collision This field indicates that the transmission was aborted after 16 successive collisions when you transmits the current packet. If MAC_Configuration[DR] = 1, this field is 1 after first collision and abort the packet transmission.
7:4	CC	Collision Count This 4-bit counter value indicates the number of collisions occurred before transmitting the packet. The count is not valid when the EC bit = 1.
3	ED	Excessive Deferral This field indicates that the transmission ends because of an excessive deferral of over 24,288 bit times (155,680 bits times in 1000 Mbit/s mode or Jumbo packet enabled mode) if MAC_Configuration[DC] = 1. When TBS is enabled in Full duplex mode and this field is 1, it indicates that the frame drops after the expiry time is reached.
2	UF	Underflow Error This field indicates that MAC aborts the packet because the data arrived late from the system memory. The underflow error can occur because of either of the following conditions: • DMA encounters an empty transmit buffer when it transmits the packet • The application filled the MTL Tx FIFO slower than the MAC transmit rate The transmission process enters the Suspended state and sets the underflow bit corresponding to a queue in MTL_Interrupt_Status.
1	DB	Deferred Bit This field indicates that MAC defers before transmitting because of presence of carrier. This field is valid only in the Half-Duplex mode.
0	IHE	IP Header Error When IP Header Error is 1, this field indicates that the checksum offload engine detects an IP header error. This field is valid only when transit checksum offload is enabled. Otherwise, it is reserved. If COE detects an IP header error, it inserts an IPv4 header checksum if the Ethernet type field indicates an IPv4 payload. In Full-Duplex mode, when EST/Qbv is enabled and this field is 1, it indicates the frame drop status due to the frame size error or schedule error.

72.16.3.2 Transmit context descriptor

The transmit context descriptor can provide any time before a packet descriptor. The context is valid for the current packet and subsequent packets. The context descriptor provide the timestamps for one-step timestamp correction and VLAN tag ID for VLAN insertion feature. Write back is done on a context descriptor only to write 0 to OWN bit.

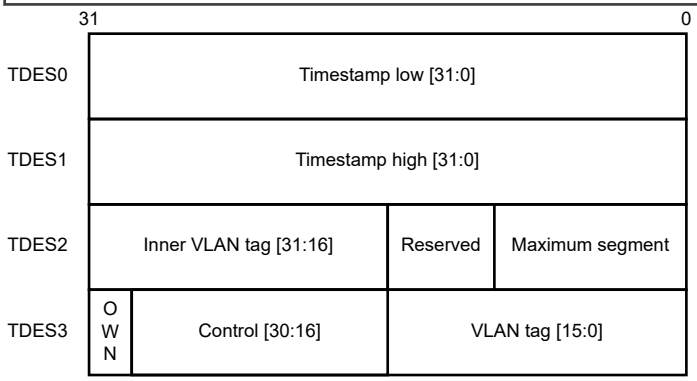
NOTE

DMA internally store the VLAN tag IDs and MSS values which the application provides in a context descriptor with their corresponding valid bits set. DMA always passes the last valid VLAN tag to the MTL, when the outer or inner VLAN tag is provided with the valid bit set. The application cannot invalidate the valid VLAN tag which DMA stores. The VLAN tag is inserted or replaced based on the control inputs provided for the packet.

The inner VLAN tag control input is used only for the next packet that immediately follows the context descriptor. The application must provide a context descriptor before the normal descriptor of each packet for which DMA must use the inner VLAN tag control input.

Figure 417 shows the transmit context descriptor format.

Figure 417. Transmit context descriptor format



72.16.3.2.1 TDES0 context descriptor

Table 514 describes the TDES0 context descriptor format.

Table 514. TDES0 context descriptor

Bit	Name	Description
31:0	TTSL	Transmit Packet Timestamp Low For one-step correction, the driver can provide the lower 32 bits of timestamp in this descriptor word. DMA uses this value as the low word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are 1.

72.16.3.2.2 TDES1 context descriptor

Table 515 describes the TDES1 context descriptor format.

Table 515. TDES1 context descriptor

Bit	Name	Description
31:0	TTSH	Transmit Packet Timestamp High For one-step correction, the driver can provide the upper 32 bits of timestamp in this descriptor. DMA uses this value as the high word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are 1.

72.16.3.2.3 TDES2 context descriptor

Table 516 shows the TDES2 context descriptor format.

Table 516. TDES2 context descriptor

Bit	Name	Description
31:16	IVT	Inner VLAN Tag When the IVLTV bit of TDES3 context descriptor is 1 and the TCMSSV and OSTC bits of TDES3 context descriptor becomes 0, it indicates that the TDES2[31:16] contains the inner VLAN tag to insert in the subsequent transmit packets.
15:14	Rsvd	Reserved
13:0	MSS	TSO split header is not supported. The value of this bit is always zero.

72.16.3.2.4 TDES3 context descriptor

Table 517. TDES3 context descriptor

31	30	29:28	27	26	25:24	23	22:18	17	16	15:0
OWN	CTXT	Rsvd	OSTC	TCMSSV	Rsvd	CDE	Rsvd	IVLTV	VLTV	VT

Table 518 shows the TDES3 context descriptor format.

Table 518. TDES3 context descriptor

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that the module DMA owns the descriptor. When this field becomes 0, it indicates that the application owns the descriptor. DMA writes 0 to this field immediately after the read.
30	CTXT	Context Type This field must set to 1'b1 for context descriptor.
29:28	Rsvd	Reserved
27	OSTC	One-Step Timestamp Correction Enable When this field is 1, it indicates that DMA performs a one-step timestamp correction with reference to the timestamp values provided in TDES0 and TDES1.
26	TCMSSV	One-Step Timestamp Correction Input or MSS Valid When this field and the OSTC field are 1, it indicates that the timestamp correction input provided in TDES0 and TDES1 is valid. When the OSTC field becomes 0 and this field and the TSE bit of TDES3 are 1 in subsequent normal descriptor, it indicates that the MSS input in TDES2 is valid.
25:24	Rsvd	Reserved

Table continues on the next page...

Table 518. TDES3 context descriptor (continued)

Bit	Name	Description
23	DE	Descriptor Error When this field is 1, it indicates that the descriptor content is incorrect. DMA writes 1 to this field during write-back when it closes the context descriptor. Descriptor errors can be: • Incorrect sequence from the context descriptor. For example, a location before the first descriptor for a packet. • All ones. • CD, LD, and FD bits = 1. **NOTE** When a descriptor error occurs due to all ones or CTXT, LD, and FD bits are 1, the transmit DMA closes the transmit descriptor with DE and LD bits are 1. When you write 1 to IOC bit in TDES2 of corresponding first descriptor, the transmit DMA writes 1 to TI bit in the DMA_CH#_Status register **NOTE** On the basis of CTXT, LD, and FD bits of the transmit descriptor, the subsequent descriptor might considered as the first descriptor (even if FD bit is not 1) and partial packet is sent.
22:20	Rsvd	Reserved
19:18	IVTIR	Inner VLAN Tag Insert or Replace When this field is 1, it requests MAC to perform inner VLAN tagging or un-tagging before transmitting the packets. If the packet is modified for VLAN tags, MAC automatically recalculates and replaces the CRC bytes. The following list describes the values of these bits: • 2'b00: Do not add the inner VLAN tag. • 2'b01: Remove the inner VLAN tag from the packets before transmission. You must use this option only with the VLAN frames. • 2'b10: Insert an inner VLAN tag with the tag value programmed in MAC_Inner_VLAN_Incl or context descriptor. • 2'b11: Replace the inner VLAN tag in packets with the tag value programmed in MAC_Inner_VLAN_Incl or context descriptor. You must use this option only with the VLAN frames.. These fields are valid when you select the enable SA and VLAN insertion on transit and enable double VLAN processing options.
17	IVLTV	Inner VLAN Tag Valid When this field is 1, it indicates that IVT field of TDES2 is valid.
16	VLTV	VLAN Tag Valid When this field is 1, it indicates that VT field of TDES3 is valid.
15:0	VT	VLAN Tag

Table continues on the next page...

Table 518. TDES3 context descriptor (continued)

Bit	Name	Description
		Contains the VLAN tag to insert or replace in the packet. This field is used as VLAN tag only when MAC_VLAN_Incl[VLTI] becomes 0.

72.16.4 Receive descriptor

DMA in the module reads a descriptor only if the tail pointer is different from the base pointer or current pointer. It is recommended to have a descriptor ring with a length that accommodates at least two complete packets which MAC receives. Otherwise, the performance of DMA is impacted because of the unavailability of the descriptors. In such situations, the RxFIFO in MTL becomes full and starts dropping packets.

The following receive descriptors are present:

- Normal descriptors
- Context descriptors

You can prepare all the receive descriptors and give to DMA as normal descriptors with the content as shown in receive normal descriptor (read format). DMA reads this descriptor and the receive DMA closes the descriptor with the corresponding packet status after transferring a received packet (or part of) to the buffers indicated by the descriptor. The receive normal descriptor (write-back format) describes the format of this status.

For some packets, only the normal descriptor bits cannot write the complete status. For such packets, the receive DMA writes the extended status to the next descriptor (without processing or using the buffers or pointers embedded in that descriptor). Receive context descriptor describes the descriptor write back format and content.

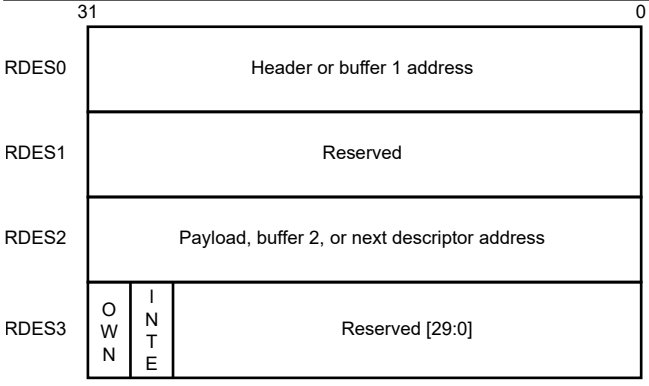
72.16.4.1 Receive normal descriptor (read format)

The read format for a receive normal descriptor is made up of the following:

- A header or buffer 1 address
- Reserved field
- Payload or buffer 2 or next descriptor address
- A 30-bit reserved field
- OWN bit
- An interrupt field

Figure 418 shows the read format for a receive normal descriptor.

Figure 418. Receive normal descriptor read format



NOTE

In the receive descriptor (read format), if the buffer address field is all zeros, the module does not transfer data to that buffer and skips to the next buffer or next descriptor.

72.16.4.1.1 RDES0 normal descriptor (read format)

Table 519 shows the RDES0 normal descriptor read format.

Table 519. RDES0 normal descriptor (read format)

Bit	Name	Description
31:0	BUF1AP	Header or Buffer 1 Address Pointer When the SPH field of control register of a channel becomes 0, these field indicates the physical address of buffer 1. When the SPH field is 1, these field indicates the physical address of header buffer where the receive DMA writes the L2/L3/L4 header bytes of the received packet. The application can program a byte-aligned address for this buffer which means that the LS bits of this field can be non-zero. However, DMA performs a write operation with RDES0[1:0] (or RDES0[2:0]/[3:0] in case of 64-/128-bit configuration) as zero, when the start of packet is transfered. However, the packet data shifts per the actual offset which the buffer address pointer provides. If the address pointer points to a buffer where the middle or last part of the packet is stored, DMA ignores the offset address and writes to the full location which the data-width indicates .

72.16.4.1.2 RDES1 normal descriptor (read format)

Table 520 shows the RDES1 normal descriptor read format.

Table 520. RDES1 normal descriptor (read format)

Bit	Name	Description
31:0	Reserved or BUF1AP	Contains the most-significant 32 bits of the buffer 1 address pointer in 64-bit Addressing mode. Otherwise, this field is reserved.

72.16.4.1.3 RDES2 normal descriptor (read format)

Table 521 shows the RDES2 normal descriptor read format.

Table 521. RDES2 normal descriptor (read format)

Bit	Name	Description
31:0	BUF2AP	Buffer 2 Address Pointer Indicates the physical address of buffer 2. When the SPH bit of the DMA_CH#_Control register = 1, it indicates that the buffer address pointer must be bus width-aligned, that is, RDES2[3:0, 2:0, or 1:0] = 0 corresponding to 128, 64, or 32 bus width. LSBs are ignored internally. When the SPH bit of the DMA_CH#_Control register becomes 0, it indicates that there is no limitations on the RDES2 value. However, the RxDMA uses the LS fields of the pointer address only when it transfers the start bytes of a packet. If the BUF2AP provides the address of a buffer in which the middle or last part of a packet is stored, DMA ignores BUF2AP[3:0 or 2:0 or 1:0] (corresponding to 128- or 64- or 32-bit data-bus) and writes to the complete location.

72.16.4.1.4 RDES3 normal descriptor (read format)

#unique_2947/unique_2947_Connect_42_id_1746ef2a-ecdf-425a-a0eb-74c5a5b90e21 describes the RDES3 normal descriptor read format.

31	30	19-26	25	24	23:0
OWN	IOC	Rsvd	BUF2V	BUF1V	Rsvd

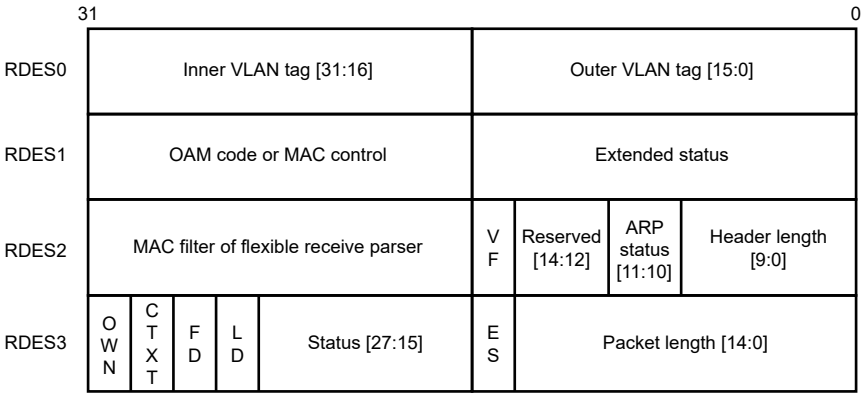
Table 522. RDES3 normal descriptor (read format)

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that the module DMA owns the descriptor. When this field becomes 0, it indicates that the application owns the descriptor. DMA write 0 this field when either of the following conditions is true: • DMA completes the packet reception. • The buffers associated with the descriptor are full.
30	IOC	Interrupt Enabled on Completion When this field is 1, it indicates that an interrupt is issued to the application when DMA closes this descriptor.
29:26	Rsvd	Reserved
25	BUF2V	Buffer 2 Address Valid When this field is 1, it indicates to the DMA that the buffer 2 address specified in RDES2 is valid. The application must write 1 to this field so that DMA can use the address, to which the buffer 2 address in RDES2 points to write received packet data.
24	BUF1V	Buffer 1 Address Valid When this field is 1, it indicates to the DMA that the buffer 1 address specified in RDES1 is valid. The application must write 1 to this field, if DMA uses the address to which the buffer 1 address in RDES1 points to write the received packet data.
23:0	Rsvd	Reserved

72.16.4.1.5 Receive normal descriptor (write-back format)

Figure 419 shows the write-back format for a receive normal descriptor.

Figure 419. Receive normal descriptor (write-back format)



NOTE

When you enables the flexible receive parser, RDES2[31:16] indicates the parser status, and not the MAC filter status. The MAC filter status is not available when flexible receive parser is enabled.

72.16.4.1.5.1 RDES0 normal descriptor (write-back format)

Table 523 shows the write-back format for the RDES0 normal descriptor.

Table 523. RDES0 normal descriptor (write-back format)

Bit	Name	Description
31:16	IVT	Inner VLAN Tag Contains the inner VLAN tag of the received packet if the RS0V bit of RDES3 = 1. This field is valid only when you enables the double VLAN tag processing and VLAN tag stripping.
15:0	OVT	Outer VLAN Tag Contains the outer VLAN tag of the received packet if the RS0V bit of RDES3 = 1.

72.16.4.1.5.2 RDES1 normal descriptor (write-back format)

The status fields in the write-back format are valid only for the last descriptor (RDES3[28] = 1). #unique_2949/unique_2949_Connect_42_table_18e086e2-bda5-4002-b444-4d7ac47c01e8 provide the details of the write-back format for RDES1 normal descriptor.

31:16	15	14	13	12	11:8	7	6	5	4	3	2:0
OPC	TD	TSA	PV	PFT	PMT	IPCE	IPCB	IPV6	IPV4	IPHE	PT

Table 524. RDES1 normal descriptor (write-back format)

Bit	Name	Description
31:16	OPC	Indicates any one of the following: • OAM sub-type code: If bits [18:16] of RDES3 as described in RDES3 normal descriptor (read format) are 111b, this field contains the OAM sub-type and code fields.

Table continues on the next page...

Table 524. RDES1 normal descriptor (write-back format) (continued)

Bit	Name	Description
		• MAC control packet opcode: Bits 15:8 of RDES3 as described in RDES3 normal descriptor (read format) contain the subtype and bits 7:0 contain the code.
15	TD	Timestamp Dropped Indicates that the timestamp was captured for this packet but it dropped in the MTL Rx FIFO because of overflow. This field is available only when you select the timestamp feature. Otherwise, this field is reserved.
14	TSA	Timestamp Available Indicates that the timestamp value is available in a context descriptor word 2 (RDES2) and word 1(RDES1) when timestamp is present. This is valid only when the last descriptor field (RDES3 [28]) = 1 You can write the context descriptor in the next descriptor just after the last normal descriptor for a packet.
13	PV	PTP Version Indicates that the received PTP message has the IEEE 1588 version 2 format. When this field becomes 0, it indicates the IEEE 1588 version 1 format. This field is available only when you select the timestamp feature. Otherwise, this field is reserved.
12	PFT	PTP Packet Type Indicates that the PTP message is sent directly over Ethernet. This field is available only when you select the timestamp feature. Otherwise, this field is reserved.
11:8	PMT	PTP Message Type Encodes to give the type of the message received: • 0000: No PTP message received • 0001: SYNC (all clock types) • 0010: Follow_Up (all clock types) • 0011: Delay_Req (all clock types) • 0100: Delay_Resp (all clock types) • 0101: Pdelay_Req (in peer-to-peer transparent clock) • 0110: Pdelay_Resp (in peer-to-peer transparent clock) • 0111: Pdelay_Resp_Follow_Up (in peer-to-peer transparent clock) • 1000: Announce • 1001: Management • 1010: Signaling • 1011–1110: Reserved • 1111: PTP packet with reserved message type These fields are available only when you select the timestamp feature.

Table continues on the next page...

Table 524. RDES1 normal descriptor (write-back format) (continued)

Bit	Name	Description
7	IPCE	IP Payload Error When this field is 1, it indicates either of the following: • The 16-bit IP payload checksum (that is, the TCP, UDP, or ICMP checksum) which MAC calculates does not match the corresponding checksum field in the received segment. • The TCP, UDP, or ICMP segment length does not match the payload length value in the IP header field. • The TCP, UDP, or ICMP segment length is less than minimum allowed segment length for TCP, UDP, or ICMP. Bit 15 (ES) of RDES3 is not set when this field is 1.
6	IPCB	IP Checksum Bypassed Indicates that the checksum offload engine is bypassed. This field is available when you select the enable receive TCP/IP checksum check feature.
5	IPV6	IPv6 header Present Indicates that an IPV6 header is detected. When you select the enable split header feature option and the SPH bit of control register of a channel is 1, the IPV6 header is available in the header buffer area to which RDES0 points.
4	IPV4	IPV4 Header Present Indicates that an IPV4 header is detected. When the SPH bit of RDES3 is 1, the IPV4 header is available in the header buffer area to which RDES0 points.
3	IPHE	IP Header Error When this field is 1, it indicates either of the following: • The 16-bit IPv4 header checksum which MAC calculates does not match the received checksum bytes. • The IP datagram version is not consistent with the Ethernet type value. • Ethernet packet does not have the expected number of IP header bytes. This field is valid when either bit 5 or bit 4 is 1. This field is available when you select the enable receive TCP/IP checksum check feature.
2:0	PT	Payload Type Indicates the type of payload encapsulated in the IP datagram that the receive checksum offload engine (COE) process. • 3'b000: Unknown type or IP/AV payload not processed • 3'b001: UDP • 3'b010: TCP • 3'b011: ICMP • 3'b110: AV tagged data packet • 3'b111: AV tagged control packet

Table continues on the next page...

Table 524. RDES1 normal descriptor (write-back format) (continued)

Bit	Name	Description
		• 3'b101: AV untagged control packet • 3'b100: IGMP, if IPV4 header present field = 1, else DCB (LLDP) control packet If the COE does not process the payload of an IP datagram because there is an IP header error or fragmented IP, it sets these fields to 3'b000.

72.16.4.1.5.3 RDES2 normal descriptor (Write-back format)

31:29	28	27	26:19	18	17	16	15	14	13:11	10	9:0
L3L4FM	L4FM	L3FM	MADRM	HF	DAF	SAF	OTS	ITS	Rsvd	ARPNR	HL

Table 525. RDES2 normal descriptor (Write-back format)

Bit	Name	Description
31:29	L3L4FM	Layer 3 and Layer 4 Filter Number Matched Indicates the number of the layer 3 and layer 4 filter that matches the received packet: • 000: Filter 0 • 001: Filter 1 • 010: Filter 2 • 011: Filter 3 • 100: Filter 4 • 101: Filter 5 • 110: Filter 6 • 111: Filter 7 This field is valid only when bit 28 or bit 27 = 1. When more than one filter matches, these fields provide the number of lowest filter. **NOTE** This status is not available when Flexible RX Parser is enabled.
28	L4FM	Layer 4 Filter Match When this field is 1, it indicates that the received packet matches one of the enabled layer 4 port number fields. This status is given only when one of these conditions is true: • Layer 3 fields are not enabled and all enabled layer 4 fields match • All enabled layer 3 and layer 4 filter fields match When more than one filter matches, this bit gives the layer 4 filter status of filter indicated by Bits[31:29].

Table continues on the next page...

Table 525. RDES2 normal descriptor (Write-back format) (continued)

Bit	Name	Description
		NOTE This status is not available when the flexible receive parser is enabled.
27	L3FM	Layer 3 Filter Match When this field is 1, it indicates that the received packet matches one of the enabled layer 3 IP address fields. This status is given only when one of these conditions is true: • All enabled layer 3 fields match and bypasses all enabled layer 4 fields. • All enabled filter fields match. When more than one filter matches, this field gives the layer 3 filter status of filter which bits[31:29] indicate. **NOTE** This status is not available when the flexible receive parser is enabled.
26:19	MADRM	MAC Address Match or Hash Value When HF becomes 0, it indicates that this field contains the MAC address register number that matches the received packet destination address. This field is valid only if the DAF field becomes 0. When HF = 1, it indicates that this field contains the hash value that MAC computes. A packet passes the hash filter when the field corresponding to the hash value is 1 in the hash filter register. **NOTE** This status is not available when the flexible receive parser is enabled.
18	HF	Hash Filter Status When this field is 1, it indicates that the packet passes the MAC address hash filter. Bits[26:19] indicate the hash value. **NOTE** This status is not available when the flexible receive parser is enabled.
17	DAF/RXPI	Destination Address Filter Fail When flexible receive parser is disabled, and this field is 1, it indicates that the packet fails the DA filter in MAC. When flexible receive parser is enabled, and this field is 1, it indicates that the packet parsing is incomplete (RXPI) due to ECC error. **NOTE** When this field is 1, RDES3 ES field is also 1.
16	SAF/RXPD	SA Address Filter Fail When flexible receive parser is disabled, and this field is 1, it indicates that the packet fails the SA filter in MAC.

Table continues on the next page...

Table 525. RDES2 normal descriptor (Write-back format) (continued)

Bit	Name	Description
		When flexible receive parser is enabled and this field is 1, it indicates that the parser drops the RXPD packet. **NOTE** When this field is 1, RDES3 ES field is also 1.
15	OTS	VLAN Filter Status When this field is 1, it indicates that the received packet VLAN tag passes the VLAN filter. This field redefines as an outer VLAN tag filter status (OTS). This field is valid for both single and double VLAN tagged frames.
14	ITS	Inner VLAN Tag Filter Status (ITS) This field is valid only for double VLAN tagged frames, when double VLAN processing is enabled. See Filter status for more information.
13:11	Rsvd	Reserved
10	ARPNR	ARP Reply Not Generated When this field is 1, it indicates that MAC did not generate the ARP reply for the ARP request packet it receives.. This field is 1 when MAC is busy transmitting ARP reply to earlier ARP request (only one ARP request is processed at a time). This field is reserved when you do not select the enable IPv4 ARP offload option.
9:0	HL	L3/L4 Header Length Contains the length of the packet header which MAC splits at L3 or L4 header boundary which the MAC receiver identifies. This field is valid only when the first descriptor bit = 1 (FD = 1). You can write the header data to the buffer 1 address of corresponding descriptor. If the header length is zero, this field is not valid. It implies that MAC does not identify and split the header. This field is valid when you select the enable split header feature option.

72.16.4.1.5.4 RDES3 normal descriptor (Write-back format)

#unique_2951/unique_2951_Connect_42_id_9cde4392-a0c2-4731-bcb2-bf9dcd789aae describes the write-back format for the RDES3 normal descriptor.

31	30	29	28	27	26	25	24	23	22	21	20	19	18:16	15	14:0
OWN	CTXT	FD	LD	RS2V	RS1V	RS0V	CE	GP	RWT	OE	RE	DE	LT	ES	PL

Table 526. RDES3 normal descriptor (Write-back format)

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that the module DMA owns the descriptor. If this field becomes 0, it indicates that the application owns the descriptor. DMA writes 0 to this field when either of these conditions is true: • DMA completes the packet reception. • The buffers associated with the descriptor are full.
30	CTXT	Receive Context Descriptor When this field is 1, it indicates that the current descriptor is a context type descriptor. DMA writes 1'b0 to this field for a normal receive descriptor. When CTXT and FD bits are used together, {CTXT, FD} • 2’b00: Intermediate descriptor • 2’b01: First descriptor • 2’b10: Reserved • 2’b11: Descriptor error (due to all 1s) **NOTE** When descriptor error occurs, the receive DMA closes the receive descriptor indicating a descriptor error. This receive descriptor is skipped and the buffer addresses are not used to write the packet data. Receive DMA will write 1 to the CDE bit in DMA_CH#_Status register but not to the RI bit even when IOC = 1, as this is not marked as the last receive descriptor for the packet. The subsequent valid receive descriptor writes the packet data.
29	FD	First Descriptor When this field is 1, it indicates that this descriptor contains the first buffer of the packet. If the size of the first buffer is 0, the second buffer contains the beginning of the packet and if the size of the second buffer is also 0, the next descriptor will contain the beginning of the packet. See the CTXT bit description for more information on how to use the CTXT bit and FD bit together.
28	LD	Last Descriptor When this field is 1, it indicates that this descriptor buffers points to the last buffers of the packet.
27	RS2V	Receive Status RDES2 Valid When this field is 1, it indicates that the status in RDES2 is valid and DMA writes it. This field is valid only when the LD bit of RDES3 = 1.
26	RS1V	Receive Status RDES1 Valid When this field is 1, it indicates that the status in RDES1 is valid and DMA writes it. This field is valid only when the LD bit of RDES3 = 1.
25	RS0V	Receive Status RDES0 Valid When this field is 1, it indicates that the status in RDES0 is valid and DMA writes it. This field is valid only when the LD bit of RDES3 = 1.

Table continues on the next page...

Table 526. RDES3 normal descriptor (Write-back format) (continued)

Bit	Name	Description
24	CE	CRC Error When this field is 1, it indicates that a cyclic redundancy check (CRC) error occurs on the received packet. This field is valid only when the LD bit of RDES3 = 1.
23	GP	Giant Packet When this field is 1, it indicates that the packet length exceeds the specified maximum Ethernet size of 1518, 1522, or 2000 bytes (9018 or 9022 bytes if jumbo packet enable = 1). **NOTE** Giant packet indicates only the packet length. It does not cause any packet truncation.
22	RWT	Receive Watchdog Timeout When this field is 1, it indicates that the receive watchdog timer expires when it receives the current packet. The current packet truncates after watchdog timeout.
21	OE	Overflow Error When this field is 1, it indicates that the received packet is damaged because of buffer overflow in receive FIFO. **NOTE** This field is 1 only when DMA transfers a partial packet to the application. This happens only when the receive FIFO operates in the Threshold mode. In the Store-and-Forward mode, all the partial packets drops completely in the receive FIFO.
20	RE	Receive Error When this field is 1, it indicates that the gmii_rxer_i signal asserts when the gmii_rxdv_i signal asserts during the packet reception. This error also includes carrier extension error in the GMII and Half-duplex mode. Error can be of less or no extension, or error (rxd!= 0f) during extension.
19	DE	Dribble Bit Error When this field is 1, it indicates that the received packet has a non-integer multiple of bytes (odd nibbles). This field is valid only in the MII mode.
18:16	LT	Length/Type Field Indicates if the packet received is a length packet or a type packet. The encoding of the 3 bits is as follows: • 3'b000: The packet is a length packet. • 3'b001: The packet is a type packet. • 3'b011: The packet is a ARP request packet type • 3'b100: The packet is a type packet with VLAN tag • 3'b101: The packet is a type packet with double VLAN tag • 3'b110: The packet is a MAC control packet type • 3'b111: The packet is a OAM packet type

Table continues on the next page...

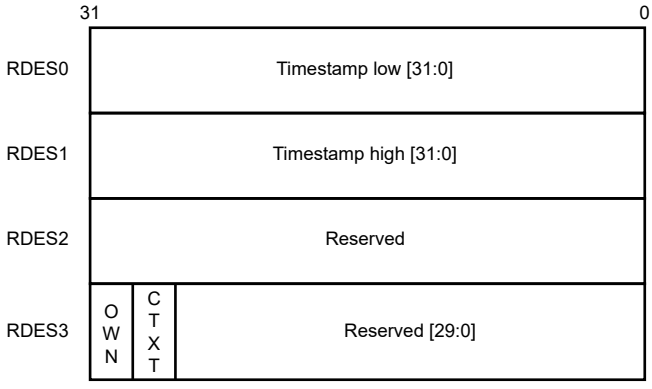
Table 526. RDES3 normal descriptor (Write-back format) (continued)

Bit	Name	Description
		• 3'b010: Reserved
15	ES	Error Summary When this field is 1, it indicates the logical OR of these bits, : • RDES3[24]: CRC error • RDES3[19]: Dribble error • RDES3[20]: Receive error • RDES3[22]: Watchdog timeout • RDES3[21]: Overflow error • RDES3[23]: Giant packet • RDES2[17]: Destination address filter fail, when the flexible receive parser is enabled • RDES2[16]: SA address filter fail, when flexible receive parser is enabled This field is valid only when the LD bit of RDES3 = 1.
14:0	PL	Packet Length Indicates the byte length of the received packet that was transferred to system memory (including CRC). This field is valid when the LD bit of RDES3 = 1 and overflow error bits becomes 0. The packet length also includes the two bytes that appends to the Ethernet packet when IP checksum calculation is enabled and the received packet is not a MAC control packet. This field is valid when the LD bit of RDES3 = 1. When the last descriptor and error summary bits are not 1, this field indicates the accumulated number of bytes that are transferred for the current packet.

72.16.4.1.6 Receive context descriptor

This descriptor is read-only for the application. Only DMA can write to this descriptor. The context descriptor provides information about the extended status related to the last received packet. Bit 30 of RDES3 indicates the context type descriptor.

Figure 420. Receive context descriptor format



72.16.4.1.6.1 RDES0 context descriptor

Table 527. RDES0 context descriptor

Bit	Name	Description
31:0	RTSL	Receive Packet Timestamp Low DMA updates this field with least significant 32-bits of the timestamp captured for the corresponding receive packet. When this field and the RTSH field of RDES1 show all-ones value, you must consider the timestamp as corrupt.

72.16.4.1.6.2 RDES1 context descriptor

Table 528. RDES1 context descriptor

Bit	Field	Description
31:0	RTSH	Receive Packet Timestamp High DMA updates this field with the most significant 32-bits of the timestamp captured for the corresponding receive packet. When this field and the RTSL field of RDES0 show all-ones value, you must consider the timestamp as corrupt.

72.16.4.1.6.3 RDES2 context descriptor

Table 529. RDES2 context descriptor

Bit	Description
31:0	Reserved

72.16.4.1.6.4 RDES3 context descriptor

31	30	29	28:0
OWN	CTXT	DE	Rsvd

Table 530. RDES3 context descriptor

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that DMA owns the descriptor. When this field becomes 0, it indicates that the application owns the descriptor. DMA clears this field when either of these conditions is true: • DMA completes the packet reception. • The buffers associated with the descriptor are full.
30	CTXT	Receive Context Descriptor

Table continues on the next page...

Table 530. RDES3 context descriptor (continued)

Bit	Name	Description
		When this field is 1, it indicates that the current descriptor is a context descriptor. DMA writes 1'b1 to this field for context descriptor. DMA writes 2'b11 to indicate a descriptor error due to all = 1. When CTXT and DE bits are used together {CTXT, DE} • 2'b00: Reserved • 2'b01: Reserved • 2'b10: Context descriptor • 2'b11: Descriptor error Note: When descriptor error occurs, the receive DMA closes the receive descriptor indicating descriptor error. This receive descriptor is skipped and the buffer addresses does not write the packet data. The receive DMA writes 1 to the CDE bit in DMA_CH#_Status register but not to the RI bit even when IOC = 1, because this is not marked as last receive descriptor for the packet. The subsequent valid receive descriptor writes the packet data.
29	DE	Descriptor Error See the CTXT bit description for more information about how to use the DE bit along with CTXT bit.
28:0	Rsvd	Reserved

72.16.5 Enhanced descriptor for time-based scheduling

This feature needs 32 bytes enhanced descriptors to enable on all the DMA channels that uses this feature (write 1 to EDSE bit of DMA_CH(#i)_TX_Control register).

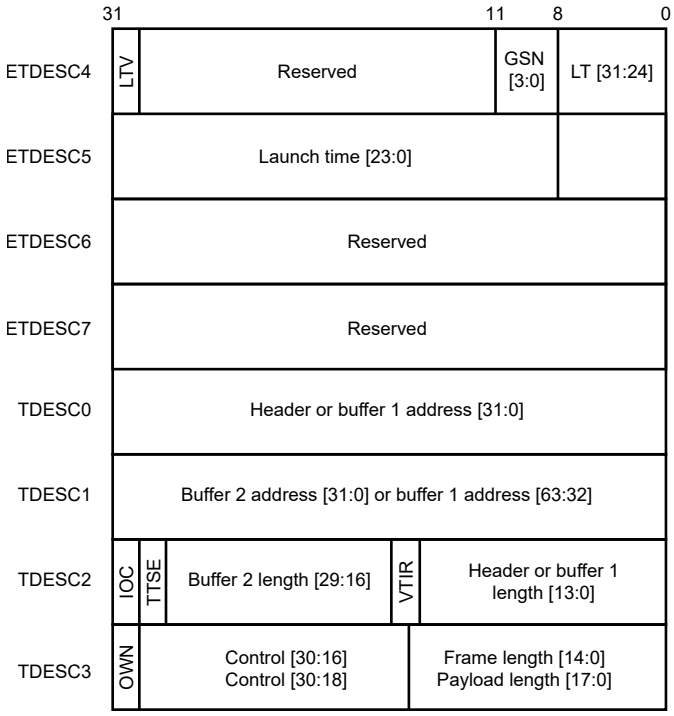
The structure of 32-byte descriptor for the context and the normal descriptor in read and write formats are described in the upcoming sections.

72.16.5.1 Enhanced normal descriptor - Read (32-bit mode)

These fields are present in the first 16 bytes of the enhanced descriptor format of normal descriptor:

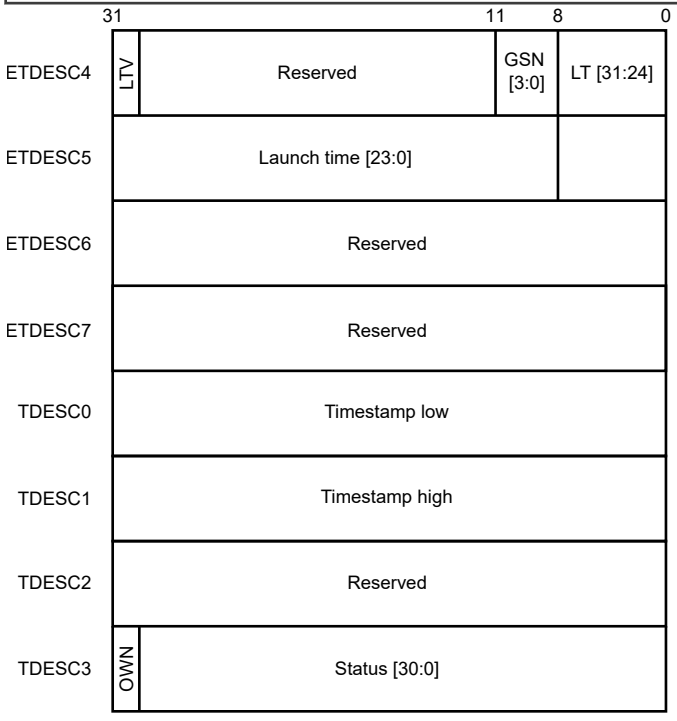
- LTV- Indicates that the launch time (LT) and GSN fields present in the descriptor are valid. The LTV must be = 1 only if the FD bit of the descriptor is not.
- GSN- Indicates the GCL slot number associated with the packet.
- LT- Indicates the launch time associated with the packet.

Figure 421. Enhanced normal descriptor - Read (32-bit mode)



72.16.5.2 Enhanced normal descriptor (Write, 32-bit mode)

Figure 422. Enhanced normal descriptor write (32-bit mode)



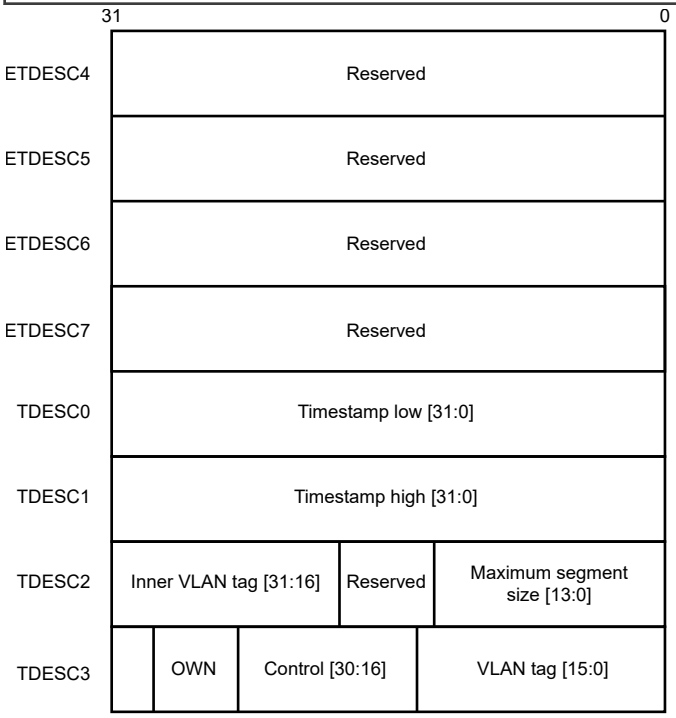
NOTE

- All the enhanced descriptors, example, assumes a 32-bit data width configurations. However, the same definitions are valid in 64-bit and 128-bit configurations.
- In both the write back formats, you cannot modify the extended 16 bytes. The remaining 16 bytes (TDESC0 to TDESC3) are written back per the previous 16 bytes descriptor format.
- When you enable the fetch time it overrides the AV slot function
- When an unaligned new GCL list (with a different CTR) is installed it is recommended not to have traffic during the installation of the new list. Any traffic during the switching of the lists might have unpredictable behavior regarding fetch, launch, and launch expiry because the CTR and BTR values get updated when the frame is processed

72.16.5.3 Enhanced context descriptor (Read, 32-bit mode)

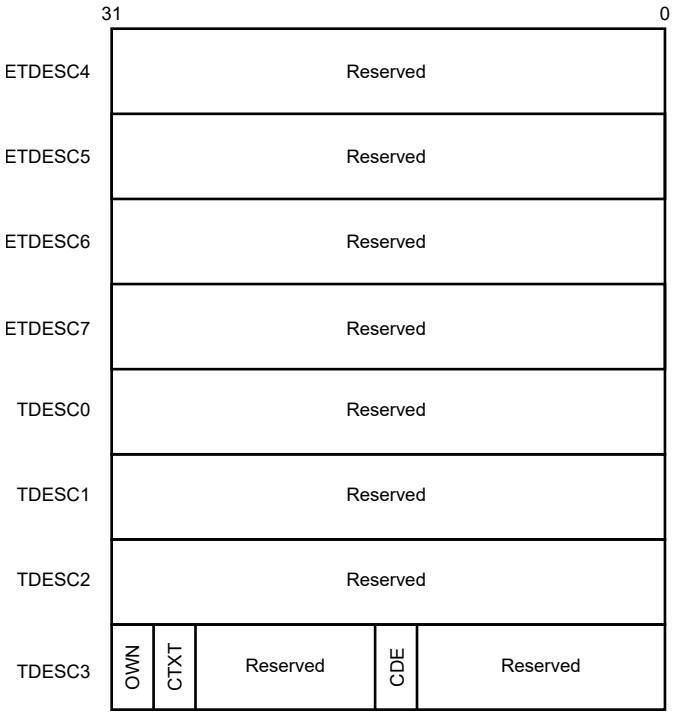
The first 16 bytes of the enhanced context descriptor (ETDESC4 to ETDESC7) are reserved and must be = 0. The fields present in the last bytes of the descriptor (TDESC0 to TDESC3) are same as the context descriptor (TDESC0 to TDESC3) in 16 byte format.

Figure 423. Enhanced context descriptor - Read



72.16.5.4 Enhanced context descriptor (Write, 32-bit mode)

Figure 424. Enhanced context descriptor - Write



72.17 Programming

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

This section provides the instructions for initializing the DMA or MAC registers in the proper sequence.

72.17.1 Initializing DMA

Perform these steps to initialize DMA:

1. Provide a software reset. This resets all the MAC internal registers and logic (bit-0 of DMA_Mode).
2. Wait for the reset process (poll bit 0 of DMA_Mode which is only clears after the reset operation completes) to complete.
3. Program these fields to initialize DMA_SysBus_Mode:
    a. DMA_SysBus_Mode[AAL]
    b. Fixed burst or undefined burst
    c. Burst mode values in case of AHB bus interface, OSR_LMT in case of AXI bus interface.
    d. Select the maximum burst length possible on the AXI bus (bits [7:1]), if fixed length value is enabled.
4. Create a descriptor list for a transmit and receive. Also, ensure that DMA (write 1 to bit 31 of descriptor TDES3/RDES3) owns the descriptors. See Descriptors for more information about descriptors.
5. Program the transmit and receive ring length registers (DMA_CH(#i)_TxDesc_Ring_Length (for i = 0; i <= 1) and DMA_CH(#i)_RxDesc_Ring_Length (for i = 0; i <= 1)). The programmed ring length must be at least 4.

NOTE

The descriptor address from the start to the end of the ring must not cross the 4 GB boundary.

6. Initialize the receive and transmit descriptor list address with the base address of the transmit and receive descriptor (DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1), DMA_CH(#i)_RxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)). Also, program the transmit and receive tail pointer registers that indicates DMA about the available descriptors (DMA_CH(#i)_TxDesc_Tail_Pointer

(for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)).

NOTE

For 40-bit or 48-bit addressing mode, program the higher address List registers (DMA_CH[n]_TxDesc_List_HAddress, DMA_CH[n]_RxDesc_List_HAddress).

The tailpointer registers must advance to the location immediately after the descriptors that are set so that DMA is aware that the additional descriptors are available.

7. Program the settings of these registers for the parameters like maximum burst-length (PBL) which DMA initiates, descriptor skip lengths, OSP in case of TxDMA, RBSZ in case of RxDMA, and so on:
   - DMA_CH(#i)_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)
   - DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)
   - DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)
8. Program the DMA_CH(#i)_Interrupt_Enable (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to enable the interrupts.
9. Write 1 to SR (bit 0) of the DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1) and ST (bit 0) of the DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to start the receive and transmit DMAs.
10. Repeat steps 4 to 9 for all the transit DMA and receive DMA channels selected in the hardware.

72.17.2 Initializing MTL registers

The transaction layer (MTL) registers must initialize to establish the transmit and receive operating modes and commands.

Perform these steps to initialize the MTL registers:

1. Program MTL_Operation_Mode[SCHALG] and MTL_Operation_Mode[RAA] to initialize the MTL operation in case of multiple transit and receive queues.
2. Program the receive queue to DMA mapping in MTL_RxQ_DMA_Map0.
3. Program these fields to initialize the mode of operation in MTL_TxQ0_Operation_Mode.
   a. MTL_TxQ0_Operation_Mode[TSF] or MTL_TxQ0_Operation_Mode[TTC] in case of Threshold mode.
   b. MTL_TxQ0_Operation_Mode[TXQEN] to value 2'b10 to enable transmit queue0.
   c. MTL_TxQ0_Operation_Mode[TQS]
4. Program these fields to initialize the mode of operation in MTL_RxQ0_Operation_Mode:
   a. MTL_RxQ0_Operation_Mode[RSF] or MTL_RxQ0_Operation_Mode[RTC] in case of Threshold mode.
   b. Flow control activation and de-activation thresholds for MTL receive FIFO (MTL_RxQ0_Operation_Mode[RFA] and MTL_RxQ0_Operation_Mode[RFD]).
   c. Error Packet and undersized good Packet forwarding enable (MTL_RxQ0_Operation_Mode[FEP] and MTL_RxQ0_Operation_Mode[FUP]).
   d. MTL_RxQ0_Operation_Mode[RQS]
5. Repeat previous two steps for all MTL transit and receive queues selected in the configuration.

72.17.3 Initializing MAC

The MAC configuration registers establish the operating mode of MAC. These registers must initialize before initializing DMA.

You can perform these MAC initialization operations after DMA initialization. If the MAC initialization completes before DMA is configured, enable the MAC receiver (last step in the following sequence) only after the DMA is active. Otherwise, the received frames fill the receive FIFO and overflow.

1. Provide the MAC address registers: MAC_Address0_High and MAC_Address0_Low. If more than one MAC address is enabled, , program the MAC addresses appropriately.

2. Program these fields to set the appropriate filters for the incoming frames in MAC_Packet_Filter:

a. Receive all

b. Promiscuous mode

c. Hash or perfect filter

d. Unicast, multicast, broadcast, and control frames filter settings

3. Program these fields for proper flow control in MAC_Q0_Tx_Flow_Ctrl:

a. Pause time and other Pause frame control bits

b. Transmit flow control bits

c. Flow control busy

4. Program MAC_Interrupt_Enable as required, and if applicable, for your configuration.

5. Program the appropriate fields in MAC_Configuration. For example Inter-packet gap when transmission and jabber disable.

6. Write 1 to MAC_Configuration[RE] and MAC_Configuration[TE] to start the MAC transmitter and receiver.

72.17.4 Performing normal receive and transmit operation

During normal operation of the module, read the normal and transmit interrupts, poll the descriptors, suspend the DMA (if it does not own descriptors), and read the values of the current host transmitter or receiver descriptor pointers for debugging.

For normal operation, perform these steps:

1. Read the interrupt status for normal transmit and receive interrupts. Then poll the descriptors, read the status of the descriptor which the host owns (either transmit or receive).

2. Set appropriate values for the descriptors, ensure that the DMA owns the transmit and receive descriptors to resume the data transmission and reception.

3. If DMA does not own the descriptors (or no descriptor is available), DMA enters into SUSPEND state. The transmission or reception can resume by freeing the descriptors and writing the descriptor tail pointer to Tx/Rx tail pointer register (DMA_CH[n]_TxDesc_Tail_Pointer and DMA_CH[n]_RxDesc_Tail_Pointer).

4. Read the current host transmitter or receiver descriptor address pointer values for the debug process (DMA_CH[n]_Current_App_TxDesc and DMA_CH[n]_Current_App_RxDesc regsiter).

5. Read the current host transmit buffer address pointer and receive buffer address pointer values for the debug process (Register DMA_CH[n]_Current_App_TxBuffer and DMA_CH[n]_Cur-rent_App_RxBuffer).

72.17.5 Stopping and starting transmission

Perform these steps to pause the transmission for some time. The steps are provided for channel 0.

1. Write 0 to bit 0 (ST) of DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to disable the transmit DMA (if applicable) .

2. Wait for any previous frame transmissions to complete. Read the appropriate fields of MTL_TxQ0_Debug (MTL_TxQ0_Debug[TRCSTS] is not 01 and MTL_TxQ0_Debug[TXQSTS] = 0) to check this.

3. Write 0 to MAC_Configuration[RE] and MAC_Configuration[TE] to disable the MAC transmitter and MAC receiver.

4. Disable the receive DMA (if applicable), after you ensure that the data in the receive FIFO transfers to the system memory (read the appropriate fields MTL_RxQ0_Debug[PRXQ] = 0 and MTL_RxQ0_Debug[RXQSTS] = 00).

5. Ensure that both the transit queue and receive queue are empty (MTL_TxQ0_Debug[TXQSTS] = 0 and MTL_RxQ0_Debug[RXQSTS] = 0).

6. Restart the operation by first starting the DMAs, and then enable the MAC transmitter and receiver.

NOTE

- Do not change the configuration (such as duplex mode, speed, port, or loop back) when the MAC is actively transmitting or receiving. You can change these parameters only when the MAC transmitter and receiver are not active.
- Similarly, do not change the DMA related configuration when transmit and receive DMA are active.

72.17.6 Programming guidelines for switching to new descriptor list in RxDMA

Switching to a new descriptor list is different in the receive DMA as compared to the transit DMA. It is permitted when the RxDMA is in SUSPEND state as described below:

- RxDMA prepares the descriptors in advance.
- If the RxDMA goes to SUSPEND state because the descriptors are not available, a major failure occurs (you are not able to free the filled-up descriptors or buffers). If this issue does not rectify immediately, frames will be lost because of an RxFIFO overflow. Therefore, you can create a new descriptor list and program the RxDMA to start using it immediately, without going into STOP state.

72.17.7 Programming guidelines for multi-channel multi-queuing

72.17.7.1 Transmit

1. Program the transmit queue size in TQS field of MTL_TxQ[n]_Operation_Mode register. Based on the value programmed in TQS field, you can determine the queue size.

In the transmit operation, the number of channels is equal to the number of the queues, because of this reason, the channel-to-queue mapping is fixed.

2. Enable the queue in TXQEN in the corresponding MTL_TxQ[n]_Operation_Mode register to use the queue. In DMA configurations, enable the ST field of DMA_CH[n]_Tx_Con-trol register and corresponding TXQEN in MTL_TxQ[n]_Operation Mode register.

3. Program the scheduling method in MTL_Operation_Mode[SCHALG].

4. Program MTL_TxQ[n]_Quantum_Weight register for DCB queue per the selected algorithm. In case of CBS algorithm in AVB queues program the MTL_TxQ[n]_ETS_Control, MTL_TxQ[n]_SendSlopeCredit, MTL_TxQ[n]_HiCredit, and MTL_TxQ[n]_LoCredit registers as required.

5. Program the MAC_TxQ_Prty_Map0 register to assign a fixed priority to the queue, if DCB is enabled and PFC function is required. You can use this assigned priority to determine if the corresponding queue must stop transmitting packet on the basis of the received PFC packet.

72.17.7.2 Receive

1. Program the receive queue size in RQS field of MTL_RxQ[n]_Operation_Mode register. Determine the size of the queue, based on the value programmed in RQS field.
2. Enable the receive queues 0 to 7 in the fields RXQ0EN to RXQ7EN in MAC_RxQ_Ctrl0 for AV or DCB. In DMA configurations, SR bit of statically or dynamically map DMA_CH[n]_Rx_Control register and the corresponding RXQ[n]_EN in MAC_RxQ_Ctrl0 is enabled.
3. Based on these packet types MAC routes the receive packets to the receive queues :
    a. AV PTP packets: Based on the programming of MAC_RxQ_Ctrl1[PTPQ].
    b. AV untagged control packets: Based on the programming of MAC_RxQ_Ctrl1[AVCPQ].

c. Data center bridging (DCB) related link layer discovery protocol (LLDP) packets. Program DCBCPQ in MAC_RxQ_Ctrl1 to indicate MAC which queue should get the DCB packets.

d. VLAN tag priority field in VLAN tagged packets: Program PSRQ7-0 of the MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 register for routing the tagged packets based on the received packets USP (user priority) field to the receive queues 0 to 7.

e. Route the AV tagged control and data packets based on PSRQ field of MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers.

NOTE

The priorities set in PSRQ7-0 must be unique.

4. If multiple receive DMA channels are enabled, you must perform the following programming for proper arbitration and mapping:

a. Program MTL_Operation_Mode[RAA] to select the arbitration algorithm to decide which RxQ is read out from the RxFIFO memory.

b. Program the MTL_RxQ[n]_Control register to decide the weights and the packet arbitration for each RxQ.

c. If you program the static mapping in MTL_RxQ_DMA_Map[n] register (RXQ[n]DADMACH becomes 0), fields RXQx2DMA and others are programmed to select the channel to map each queue.

d. Write 1 to RXQ[n]DADMACH field in MTL_RxQ_DMA_Map0 to select the dynamic mapping of packets in each receive queue.

e. In dynamic channel mapping, the value of DCS field in the lowest MAC address registerthe decides the routing of a packet to a specific RxDMA channel.

72.17.7.2.1 Programming guidelines for recovering from DMA channel failure

When the DMA channel issues a bus error, follow these steps to recover from the failure.

Recovering from the receive DMA channel failure.

Perform these steps if you get bus error in the receive DMA channel:

1. Write 1 to DMA_CH0_Rx_Control[RPF]. This flushes all the packets one after the other.

This step is optional. However, writing 1 to this field prevents head-of-line blocking in the receive queues when packets sent to the RXDMA are stopped due to the bus error.

2. Re-program the specific registers of the DMA channel.

3. Start the DMA channel.

Recovering from the transmit DMA channel failure.

1. Stop the specific DMA channel, even if it is in active state.

2. Flush the corresponding MTL queue.

3. Re-program the specific registers of the DMA channel.

4. Start the DMA channel.

NOTE

Due to the known limitations in the design, reprogramming the DMA channel registers might not be always successful in recovering from a bus error. If the module is not fully functional after reprogramming the DMA, you can issue a soft reset to recover from the bus error.

72.17.8 Programming guidelines for GMII link state transitions

72.17.8.1 Transmit and receive clocks running when link down

Perform these steps when the link is down but the transmit and receive clocks are running:

NOTE

The steps are provided for channel 0.

1. Write 0 to bit 0 (ST) of DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to disable the transmit DMA (if applicable).

2. Write 0 to MAC_Configuration[RE] to disable the MAC receiver.

3. Wait for any previous frame transmissions to complete. You can check this by reading the appropriate fields of MTL_TxQ0_Debug (MTL_TxQ0_Debug[TRCSTS] is not 01) or, flush the Tx FIFO for faster empty operation.

4. Write 0 to MAC_Configuration[TE] to disable the MAC transmitter.

5. Make sure that both the transit queue and receive queue are empty (MTL_TxQ0_Debug[TXQSTS] and MTL_RxQ0_Debug[RXQSTS] are 0).

6. Read the PHY registers to know the latest configuration and accordingly program the MAC registers, after the link is up.

7. Start the transit DMA to restart the operation, and then enable the MAC transmitter and receiver.

You do not disable the receive DMA because the receiver is disabled and the FIFO does not get any data in the receive FIFO.

72.17.8.2 Transmit and receive clocks stopped when link down

Perform these steps when the link is down and the transmit and receive clocks are stopped:

NOTE

The steps are provided for channel 0.

1. Write 0 to MAC_Configuration[RE] and MAC_Configuration[TE] to disable the MAC transmitter and receiver. This does not take effect immediately as the clocks are absent.

2. Wait until the link is up and the clocks are restored.

3. Wait for the transfer of any partial frame, if any to complete at time of stopping of the transmit or receive clock. To check this, read MAC_Debug (should be all-zero). Some old packets may still remain in the TXFIFO because the MAC transmitter is stopped.

4. Read the PHY registers to know the latest operating mode and accordingly program the MAC registers.

5. Write 1 to MAC_Configuration[RE] and MAC_Configuration[TE] to restart the MAC transmitter and receiver.

72.17.9 Programming guidelines for IEEE 1588 timestamping

72.17.9.1 Initialization guidelines for system time generation

Write 1 MAC_Timestamp_Control[TSENA] to enable the timestamp feature. However, it is essential to initialize the timestamp counter after MAC_Timestamp_Control[TSENA] = 1. Perform these steps during the module initialization:

1. Write 0 to the bit 16 of MAC_Interrupt_Enable to mask the timestamp trigger interrupt.
2. Write 1 to MAC_Timestamp_Control[TSENA] to enable timestamping.
3. Program MAC_Sub_Second_Increment based on the PTP clock frequency.
4. Program MAC_Timestamp_Addend and write 1 to MAC_Timestamp_Control[TSADDREG], if you are using the fine correction approach.
5. Poll MAC_Timestamp_Control until MAC_Timestamp_Control[TSADDREG] = 0.
6. Program MAC_Timestamp_Control[TSCFUPDT] to select the fine update method (if required).

7. Program MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update with the appropriate time value.
8. Write 1 to MAC_Timestamp_Control[TSINIT].
9. Initialize the timestamp counter with the value written in the timestamp update registers so that the timestamp counter can start operation. If one-step timestamping is enabled
    a. Program Bit 27 of the TDES3 context descriptor, to enable one-step timestamping.
    b. Program MAC_Timestamp_Ingress_Asym_Corr and MAC_Timestamp_Egress_Asym_Corr to update the correction field in PDelay_Req PTP messages.
10. Enable the MAC receiver and transmitter for proper timestamping.

NOTE

If the timestamp operation is disabled by writing 0 to MAC_Timestamp_Control[TSENA], repeat all these steps to restart the timestamp operation.

72.17.9.2 System time correction

72.17.9.2.1 Coarse correction method

Perform these steps to synchronize or update the system time in one process (coarse correction method):

1. Set the offset (positive or negative) in the timestamp registers that is MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update.

2. Write 1 to MAC_Timestamp_Control[TSUPDT].

The value in the timestamp update registers is added to or subtracted from the system time when MAC_Timestamp_Control[TSUPDT] = 0.

72.17.9.2.2 Fine correction method

Perform these steps to synchronize or update the system time to reduce system-time jitter (fine correction method):

1. Calculate the rate by which you want to increment the system time slower or faster, with the help of the algorithm.

2. Update MAC_Timestamp_Addend with the new value and write 1 to MAC_Timestamp_Control[TSADDREG].

3. Wait for the time for which you want the new value of the addend register to be active. You can do this by enabling the timestamp trigger interrupt after the system time reaches the target value.

4. Program the required target time in MAC_PPS[n]_Target_Time_Seconds register and MAC_PPS[n]_Target_Time_Nanoseconds register.

5. Enable the timestamp interrupt in MAC_Interrupt_Enable[TSIE].

6. Write 1 to bit 4 in MAC_Timestamp_Control.

7. Read MAC_Interrupt_Status when this trigger causes an interrupt.

8. Reprogram MAC_Timestamp_Addend with the old value and write 1 to MAC_Timestamp_Control[TSADDREG] again.

72.17.10 Programming guidelines for AV feature

After you enable the AV feature in the module controller, follow these programming tasks:

- Initializing the DMA for QOS-AHB configurations only
- Enabling slot number checking
- Enabling average bits per slot reporting
- Disabling flow control for AV-enabled queues (transmit and receive flow control)

72.17.10.1 Initializing the DMA in audio video feature

The first step to program the AV feature in a QOS-AHB configuration is to initialize the DMA.

Use this initialization sequence for QOS-AHB/QOS-AXI/QOS-AXI4 configurations with AV feature.

1. Provide a software reset to reset all the QOS internal registers and logic (DMA_Mode[SWR]).

2. Wait for the reset process to complete. Poll DMA_Mode[SWR], which clears only after the reset operation completes.

3. Set the values in DMA_Mode to program the fields to initialize the DMA register.

4. Create a descriptor list for transmit and receive. In addition, ensure that the DMA owns the transmit and receive descriptors. When you use OSF mode, at least two transmit descriptors are required.

See Descriptors for more information about descriptors.

5. Ensure that you create three or more different transmit or receive descriptors in the list before reusing any descriptors.

6. Program the transmit and receive ring length registers (DMA_CH(#i)_TxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)). The ring length programmed must be at least 4.

7. Initialize receive and transmit descriptor list address with the base address of the transmit and receive descriptor (DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1), DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)). In addition, you must program the transmit and receive tail pointer registers to indicate the DMA about the available descriptors (DMA_CH(#i)_TxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_D-MA_TX_CH-1) and DMA_CH(#i)_RxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_D-MA_RX_CH-1)).

8. Program these fields to initialize the mode of operation in MTL_TxQ0_Operation_Mode:

a. MTL_TxQ0_Operation_Mode[TSF]

b. MTL_TxQ0_Operation_Mode[TTC]

c. MTL_TxQ0_Operation_Mode[TXQEN] to value 2'b10 to enable transmit queue0

d. MTL_TxQ0_Operation_Mode[TQS]

9. Enable the interrupts by programming DMA_CH(#i)_Interrupt_Enable (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register.

10. Repeat steps 4 through 9 for all the additional channels of AV feature.

11. Program the AV queues CBS control register, idleSlope, sendSlope, hiCredit, and loCredit registers.

12. Write 1 to bit 0 of DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1) register to start the receive and transmit DMA.

72.17.10.2 Enabling slot number checking

You can enable slot number checking feature if you specifies the intervals at which the DMA channels map to AV queues fetch the frames from the AHB/AXI system bus.

You must complete these steps after step 11 and before step 12 of Initializing the DMA in audio video feature .

You can use the slot number check feature to specify the intervals at which the DMA channels map to AV queues fetch the frames from the AHB/AXI system bus. This feature is useful for an uniform and periodic transfer of the AV traffic from the host memory and it is available only when you enable timestamping and programs MAC_Sub_Second_Increment. Perform these steps to enable the slot number checking:

1. Follow the steps described in Initialization guidelines for system time generation to enable timestamping.

2. Ensure that the SLOTNUM field (bits 22:19) of TDES3 normal descriptor (Read format) contains a valid slot number. You can read the current reference slot number from the DMA_CH(#i)_Slot_Func-tion_Control_Status (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register.

3. Write 1 to bit 0 (ESC) of the slot function control and status register of a channel to enable the slot number checking.

72.17.10.3 Enabling average bits per slot reporting

You can enable the reporting of average bits transmitted in a slot.

The CBS status register of the additional AV channels provide information about the average bits that are transmitted in a slot. You can asynchronously read this register to retrieve information about the average bits transmitted per slot. Perform these steps to enable average bits per slot reporting:

1. Follow the steps as described in the Initialization guidelines for system time generation to enable timestamping.

2. Program SLC bits [6:4], of the MTL_TxQ[n]_ETS_Control register of a channel with the number of slots over which the average transmitted bits per slot are computed.

3. Enable bit 9(ABPSSIE) of the MTL_Q[n]_Interrupt_Control_Status register of a channel to generate the average bits per slot interrupt.

NOTE

- The frequency of this interrupt depends on the value programmed in step 2. For example, when you program value 0 in the SLC field, the interrupt generates at every 125 microsecond.
- You can disable this interrupt to stop the interrupt flooding, when it is not required.

4. Read ABS bits [16:0], of the MTL_TxQ[n]_ETS_Status register of a channel on each interrupt.

NOTE

You can read the ABS fields in the polling mode even if ABPSIE bit is not enabled. When bit 1 (ABPSIS) of the MTL_TxQ[n]_ETS_Status register is 1, it indicates that a new value is updated in the ABS field.

72.17.10.4 Disabling flow control for AV enabled queues

72.17.10.4.1 Transmit flow

Program the EHFC (Enable Hardware Flow Control) field of the corresponding receive queue's MTL_RxQ[n]_Operation_Mode register to 0.

72.17.10.4.2 Receive flow control

Program PSTQ[n] field, corresponding to AV enabled transit queue in MAC_TxQ_Prty_Map0/1 register to 0.

72.17.10.5 Programming guidelines for flexible pulse-per-second output

After you enable the flexible pulse-per-second output feature in the module controller, you can perform these tasks:

- Generating single pulse on PPS
- Generating next pulse on PPS
- Generating a pulse train on PPS
- Generating an interrupt without affecting the PPS

72.17.10.5.1 Generating single pulse on PPS

Perform these steps to generate single pulse on PPS:

1. Program 11 or 10 (for interrupt) in TRGTMODSEL bits [6:5], of MAC_PPS_Control. This instructs MAC to use the target time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds) for a start time of the PPS signal output.
2. Program the start time value in the target time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds).
3. Program the PPS signal output width in MAC_PPS(#i)_Width (for i = 0; i <= 3) register.

4. Program PPSCMD bits [3:0], of MAC_PPS_Control to 0001. This instructs MAC to generate single pulse on the PPS signal output at the time programmed in the target time registers.

72.17.10.5.2 Generating next pulse on PPS

When the PPSCMD is executed (PPSCMD bits = 0), you can give the cancel start command (PPSCMD=0011) before the programmed start time elapses to cancel the pulse generation. You can also program the behavior of the next pulse in advance.

Follow these steps to program the next pulse:

1. Program the start time for the next pulse in the target time registers. This time must be more than the time at which the falling edge occurs for the previous pulse.

2. Program the next PPS signal output width in MAC_PPS(#i)_Width (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) register.

3. Program PPSCMD bits [3:0], of MAC_PPS_Control to generate a single pulse after the time at which the previous pulse is de-asserted. This instructs MAC to generate single pulse on the PPS signal output, at the time programmed in target time registers.

If you give this command before the previous pulse becomes low, then the new command overwrites the previous command and the QOS may generate only 1 extended pulse.

72.17.10.5.3 Generating a pulse train on PPS

Perform these steps to generate a pulse train on PPS:

1. Program 11 or 10 (for interrupt) in TRGTMODSEL bits [6:5] , of MAC_PPS_Control. This instructs MAC to use the target time registers for the start time of the PPS signal output.

2. Program the start time value in the target time registers.

3. Program the interval value between the train of pulses on the PPS signal output in MAC_PPS(#i)_Interval (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) register.

4. Program the PPS signal output width in MAC_PPS(#i)_Width (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) register.

5. Program PPSCMD bits [3:0], of MAC_PPS_Control to 0010. This instructs MAC to generate train of pulses on the PPS signal output with start time programmed in target time registers.

By default, the PPS pulse train is free-running unless the STOP pulse train at time or STOP pulse train immediately commands stop it.

6. Program the stop value in the target time registers. Ensure that bit 31 (TSTRBUSY) of MAC_PPS(#i)_Target_Time_Nanoseconds (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) register becomes 0 before programming the target time registers again.

7. Program PPSCMD field [bit 3:0] of MAC_PPS_Control to 0100. This stops the train of pulses on the PPS signal output after the programmed stop time specified in step 6 elapses.

You can program 0101 in PPSCMD field [bit 3:0] of MAC_PPS_Control to stop the pulse train at any time. Similarly, you can program 0110 in PPSCMD field [bit 3:0] of MAC_PPS_Control before the time (programmed in step 6) elapses to cancel the stop pulse train command (given in Step 7). Also, program 0011 in PPSCMD field [bit 3:0] of MAC_PPS_Control before the programmed start time (in Step 2) elapses to cancel the pulse train generation.

72.17.10.5.4 Generating an interrupt without affecting the PPS

MAC_PPS_Control TRGTMODSEL bits [6:5] enables you to program the target time registers to perform any one of these actions:

- Generate only interrupts.
- Generate interrupts and the PPS start and stop time.
- Generate only PPS start and stop time.

Perform these steps to program the target time registers to generate only interrupt event:

1. Program 00 (for interrupt) in MAC_PPS_Control TRGTMODSEL bits [6:5]. This instructs the MAC to use the target time registers for target time interrupt.

2. Program a target time value in the target time registers. This instructs the MAC to generate an interrupt when the target time elapses.

If TRGTMODSEL bits [6:5], are changed (for example, to control the PPS) then over-write the interrupt generation with the new mode and new programmed target time register value.

72.17.10.6 Programming guidelines for VLAN filtering on receive

Perform these steps to program VLAN filtering on receive:

1. Program MAC_VLAN_Tag register for the following fields to select the filtering method:

a. ETV: Enables 12-Bit VLAN tag comparison or 16-bit VLAN tag comparison.

b. VTHM: VLAN tag hash table match enable.

c. ERIVLT: Enables inner VLAN tag or outer VLAN Tag (to enable the inner or outer VLAN tag filtering, writing 1 to MAC_VLAN_Tag_Ctrl[EDVLP] enables the double VLAN processing).

d. ERSVLM: Enables receive S-VLAN match or C-VLAN match (write 1 to MAC_VLAN_Tag_Ctrl[ESVL] to enable S-VLAN processing).

e. DOVLTC: Ignores VLAN type for tag match

f. VTIM: Enables VLAN tag inverse match instead of the normal VLAN tag matching

2. Program VL of MAC_VLAN_Tag register for the 12-bit or 16-bit VLAN tag.

3. Program MAC_VLAN_Hash_Table, if hash filtering of VLAN tag is enabled. When MAC_VLAN_Tag_Ctrl[ETV] becomes 0, it indicates that the upper 4 bits of the calculated CRC-32 of VLAN tag are inverted and index the content of MAC_VLAN_Hash_Table. When MAC_VLAN_Tag_Ctrl[ETV] is 1, it indicates that the upper 4 bits of calculated CRC-32 of VLAN Tag index the content of MAC_VLAN_Hash_Table. For example, when MAC_VLAN_Tag_Ctrl[ETV] is 1, a hash value of 4b'1000 selects bit 8 of MAC_VLAN_Hash_Table. When MAC_VLAN_Tag_Ctrl[ETV] becomes 0 a hash value of 4'b1000 selects bit 7 of MAC_VLAN_Hash_Table.

72.17.10.7 Programming guidelines for extended VLAN filtering and routing on receive

Perform these steps, for the indirect access of the per VLAN tag registers:

- Write
  - Write the required data into MAC_VLAN_Tag_Data.
  - Program MAC_VLAN_Tag_Ctrl[OFS] with the required filter register's offset and command type to MAC_VLAN_Tag_Ctrl[CT]. For a write command, set this field to 0.
  - Write 1 to MAC_VLAN_Tag_Ctrl[OB] and wait till it becomes 0 to do the next write. This will guarantee that you have programmed the appropriate VLAN Tag Filter register.
- Read
  - Program MAC_VLAN_Tag_Ctrl[OFS] with the required register's offset and command type to MAC_VLAN_Tag_Ctrl[CT]. For a read command, set this field to 1.
  - Write 1 to MAC_VLAN_Tag_Ctrl[OB] and wait till it becomes 0. The appropriate value of the VLAN Tag Filter register is available in MAC_VLAN_Tag_Data.

72.17.10.8 Programming sequence for queue/channel based VLAN inclusion register

Perform these steps to program the queue/channel-based VLAN inclusion register:

NOTE

When MAC_VLAN_Incl[CBTI] = 1 you cannot access the indirect VLAN include registers.

1. Write 1 to MAC_VLAN_Incl[CBTI], to enable queue or channel based VLAN tag insertion on all the transmitted packets. This field must be 1 before any indirect access to the queue or channel specific MAC_VLAN_Incl(#i) register.

2. Program the VLAN tag and VLAN type to insert in packets from a particular queue or channel in MAC_VLAN_Incl[VLT] and MAC_VLAN_Incl[CSVL], corresponding offset address in ADDR field (0 for queue/channel 0, 1 for queue/channel 1, and so on) must be set. Write 0 to MAC_VLAN_Incl[RDWR] to indicate write access. The write to byte 0 (byte 3 in Big Endian mode) of MAC_VLAN_Incl initiates access to indirect access MAC_VLAN_Incl(#i) register.

3. Write 1 to MAC_VLAN_Incl[BUSY] via the module to indicate the progress of access to indirect access MAC_VLAN_Incl(#i) register. After the access completes, MAC_VLAN_Incl[BUSY] = 0. The application must not attempt subsequent access to MAC_VLAN_Incl(#i) register when the MAC_VLAN_Incl[BUSY] is 1.

4. Repeat step 2 and step 3 to program VLAN tag and VLAN type to insert in packets from the remaining queues or channels. The application must ensure that the required VLAN tag and VLAN type for all the queues or channels are programmed, otherwise an unintended VLAN tag and VLAN type might be inserted.

72.17.11 Programming guidelines for EST

Program the gate control values and time intervals in MTL_EST_Status[SWOL] along with the other EST related registers that are described in EMAC register descriptions to appropriate values. The upcoming sub-sections provide step by step details for programming the GCL and the other EST related registers.

72.17.11.1 Programming the GCL and GCL linked registers

Perform these steps to program the GCL and the four other registers implemented per GCL:

1. Access the GCL and the four other GCL-linked registers via indirect addressing using MTL_EST_GCL_Control and MTL_EST_GCL_Data. MTL_EST_Status[SWOL] indicates whether the software owns GCL0 or GCL1.
2. Write the 32-bit write data to MTL_EST_GCL_Data, to program the GCL. Then program MTL_EST_GCL_Control to write the write address and other control information.
3. In MTL_EST_GCL_Data, write data consists of up to 8 bits (configurable) of gate controls and up to 24 bits (configurable) of time interval. Programming a 0 indicates gate close and programming a 1 indicates gate open. For a 4-TC and 20-bit time interval configuration, the data width is 24-bits and the remaining 8-bits are reserved/read-only. You must write the data in the following format.

   {8'h0, TC3, TC2, TC1, TC0, 20-bit Time Interval} where TCx = 0 or 1.
4. Program MTL_EST_GCL_Control[SRWO] to 1 (to start a Write Op) and program the address and R/W fields appropriately.
5. Poll and check that the hardware clears MTL_EST_GCL_Control[SRWO] to indicate that the previous operation completes before initiating a new read-write operation via the same indirect addressing mode.
6. Repeat steps 3, 4, 5 until the programming of the GCL completes.
7. Program the BTR, CTR, TER and LLR registers, using the same indirect addressing method as described above. Write 1 to MTL_EST_GCL_Control[GCRR] appropriately. MTL_EST_GCL_Control[GCRR] interprets the address field as belonging to these registers (instead of the GCL).

After the GCL and the related registers are programmed, program MTL_EST_GCL_Control to allow hardware to own and process the GCL. When the list length (as indicated in LLR) is 1, the associated time interval must be smaller than the cycle time register value. Otherwise, an error is reported (as described in the Error handling section) as a single set of gate controls add no value in the TSN context.

NOTE

The time unit in all the GCL related registers is seconds and nanoseconds. In cases where internally generated PTP system time is used, you must program the nanoseconds field to use the Digital Rollover mode. (The value of MAC_Timestamp_Control[TSCTRLSSR]must be 1)

72.17.11.2 Programming the EST registers

After the steps mentioned in "Programming the GCL and GCL Linked Registers" completes, program MTL_EST_Control

1. Write 1 to MTL_EST_Control[CTOV] and MTL_EST_Control[TILS]. Also, write 1 to MTL_EST_Control[EEST] and MTL_EST_Control[SWOL].

   NOTE

   The CTOV recommendations for GMII for SPRAM configurations are:

   - 96 * Tx clock period, for 32 and 64 bit data width configurations.
   - 128 * Tx clock period for 128 bit data width configurations.

   The CTOV recommendations for GMII for non-SPRAM configurations are:

   - 30 * Tx clock period, when SA/VLAN insertion is enabled.
   - 22 * Tx clock period, when SA/VLAN insertion is not enabled.

2. Enables the hardware to own and process the new GCL and make a switch to the new GCL at the BTR value. The hardware provides an interrupt (if enabled) when the switch to the new list happens.
3. Performs an appropriate action to address any other interrupts (explained in the Inter-rupts section) received during the hardware execution of the GCL.

Write 1 to MTL_EST_Control[SSWL] to handoff to hardware. Software is not allowed to write to the GCL and GCL linked registers when MTL_EST_Control[SSWL] = 1, because the hardware might be using the new GCL.

The hardware resets or clears MTL_EST_Control[SSWL] when it successfully switches to the new list. The hardware also flips MTL_EST_Status[SWOL] to indicate the new GCL that the software owns.

Program the GCL that software owns (MTL_EST_Status[SWOL] indicates) as described in "Programming the GCL and GCL Linked Registers" and then program MTL_EST_Control as described above, to install a new GCL. Ensure that the new BTR is set to an appropriate value to avoid BTR Error that may require software intervention in some cases.

The packet length (frame size) information must be available at all times, to avoid transmission overruns. Therefore, in the DMA configurations, program the packet length in the first descriptor of every transmit frame. Similarly, in the MTL configuration, provide the packet length in the control word.

NOTE

The packet length provided in the transmit descriptor must account for the SA and VLAN insertion, to avoid transmission overruns, if applicable, that is, packet length must represent the actual packet length on the Ethernet line.

72.17.12 Programming the launch time in time-based scheduling

Configure the launch time in the enhanced normal transmit descriptors in DMA configurations and is driven as a control word in MTL configurations as follows:

The OSTC and launch time features are mutually exclusive and must not be used together. In case of a conflict (if OSTC = LTV = 1 in MTL configuration), give priority to OSTC and ignore the launch time.

In DMA configuration, if a context descriptor is received with a valid OSTC values immediately before receiving a first normal descriptor with LTV = 1, then the LTV is ignored.

72.17.13 Programming guidelines for media clock generation and recovery

72.17.13.1 Programming guidelines for media clock generation

These are the guidelines for the media clock generation:

1. Program the appropriate presentation time control (supported generation modes "1001-1011") to PPSCTRL_PPSCMD (for 0th instance)/PPSCMD#i (for 1,2,3 instances) of MAC_PPS_Control, to define the PPS instance in Media clock generation mode.

2. Based on the selected PPS instance, application must drive the appropriate trigger signal to the corresponding mcg_pst_trig_i[#i].

3. Based on the programmed mode, DUT captures the timestamp and programs it in the MAC_PPS(#i)_Target_Time_Seconds register. Then, mcgr_dma_req_o[#i] is set.

4. For every request which DUT generates, the module reads the corresponding MAC_PPS(#i)_Target_Time_Seconds register and asserts the corresponding mcgr_d-ma_ack_i[#i] to acknowledge the read request.

72.17.13.2 Programming guidelines for media clock recovery

These are the guidelines for the media clock recovery:

- Writing 1 to MAC_Timestamp_Control[PTGE] enables the CPT counter. In addition to configuring the initialization values for the system time, update MAC_Presn_Time_Updt with the equivalent presentation time initialization value, then MAC_Timestamp_Control[TSINIT] = 1.
- Use the increment values for the system time and also for the CPT because the increment value is in sub-seconds and sub-nanoseconds.
- Update MAC_Presn_Time_Updt with the equivalent presentation time update value (32 bit ns), when an update is requred, and also configure the updated values for the system time. Finally, MAC_Timestamp_Control[TSUPDT] must be 1 for the update.
- Program the MCGREN#i field of MAC_PPS_Control to enable the PPS instance to operate in MCGR mode.
- Program the appropriate presentation time control (supported recovery modes "0001 - 0011") to PPSCTRL_PPSCMD (for 0th instance) or PPSCMD#i (for 1,2,3 instances) of MAC_PPS_Control register, to set the PPS instance in Media clock recovery mode.
- After the PPS instance is set in Recovery mode, DUT sets the corresponding mcgr_dma_req_o[#i]. For every request which DUT generates, program the target presentation time into the MAC_PPS(#i)_Target_Time_Seconds register and assert the corresponding mcgr_dma_ack_i[#i] to acknowledge the request. Acknowledgment can be given before/after programming the Target presentation time.
- Observe the recovered media clock on the corresponding ptp_pps_o[#i].

72.17.14 Programming guidelines for ECC protection for memories

These are the guidelines for the ECC protection for memories:

- Write 1 to the appropriate field of MTL_ECC_Control to enable the ECC features for the respective memory.
- Generate a correctable interrupt (sbd_sfty_ce_intr_o) or an uncorrectable interrupt (sbd_sfty_ue_intr_o) to indicate the application, if any correctable, uncorrectable, or address errors are detected. DMA_Interrupt_Status or MTL_ECC_Interrupt_Status indicates an appropriate status.
- Provide debug mode for each memory to specify errors.

NOTE

Enable the ECC feature before the traffic is online (or after the reset), otherwise the false interrupts may trigger.

72.17.14.1 Programming guidelines for ECC hardware error injection (Debug mode)

Follow these steps for ECC hardware error injection:

- Write 1 to the appropriate field in MTL_DBG_CTL to enable the ECC error injection feature for MTL TX/RX and DMA TSO.
- Write 1 to the appropriate field in MTL_EST_GCL_Control to enable the ECC error injection feature for the EST memory.
- Write 1 to the appropriate field in MTL_RXP_Indirect_Acc_Control_Status to enable the ECC error injection for the receive parser memory. To access the receive parser memory in Debug mode, write 0 to MTL_RXP_Indirect_Acc_Control_Status[FRPE] to disable the receive parser feature.
- Ensure that all the CSR writes corresponding to one memory write must have the same value for the error injection control word, where multiple CSR writes are required for writing single data word into the memory.

72.17.15 Programming guidelines for on-chip datapath parity protection

Follow these steps for on-chip datapath parity protection:

- Writing 1 to MTL_DPP_Control[EDPP] enables the parity generation and parity detection mismatch on datapath.
- Generates an uncorrectable interrupt (sbd_sfty_ue_intr_o) to indicate the application if any parity mismatch is detected. DMA_Safety_Interrupt_Status or MTL_Safety_Interrupt_Status and MAC_DPP_FSM_Interrupt_Status indicates the appropriate status.
- Supports Debug mode for each parity generator to insert an error.

NOTE

Enable the datapath parity protection before the receive or transit traffic starts, to avoid the false safety interrupts.

72.17.16 Programming guidelines for FSM parity and timeout

Follow these steps to configure the FSM parity and timeout:

- Write 1 to MAC_FSM_Control[PRTYEN] and MAC_FSM_Control[TMOUTEN] to enable the FSM parity and timeout feature respectively.
- Force the FSMs to enter into a state with even number of 1s and wait for the safety uncorrectable interrupt to define with MAC_DPP_FSM_Interrupt_Status[FSMPES] for a parity error detection.
- Write 1 to the parity error injection using the [23:16] bits of MAC_FSM_Control to select the appropriate clock domain, for Error injection mode in FSM parity. Wait for the safety uncorrectable interrupt to be defined with MAC_DPP_FSM_Interrupt_Status[FSMPES].
- Configure the large and normal mode values which MAC_FSM_ACT_Timer[LTMRMD] and MAC_FSM_ACT_Timer[NTMRMD] indicates , for FSM timeouts.
- Use the [31:24] bits of MAC_FSM_Control to select the large or normal mode tic generation per clock domain.
- Write 1 to MAC_FSM_ACT_Timer[TMR] with the appropriate number of CSR clock cycles to generate 1us tic.
- Force the FSM in a particular clock domain to remain in an active state for timeout duration (for example, T-2T). You can check the safety interrupt and the status field for the corresponding clock domain that sets in the [15:8] bits of the MAC_DPP_FSM_Interrupt_Status .
- Select the appropriate clock domain for which timeout error injection must be set using the [15:8] bits of MAC_FSM_Control, for Error injection mode in FSM timeout. Wait for the safety uncorrectable interrupt to define with the corresponding error status in the MAC_DPP_FSM_Interrupt_Status .
- Use only normal mode tic generation, for application or CSR interface timeout. On the basis of the time-outs of configured interface, appropriate fields (MSTTES, SLVTES) of MAC_DPP_FSM_Interrupt_Status is defined with the safety interrupt.

72.18 EMAC register descriptions

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

NOTE

This chip has 16 KB allocated per Ethernet controller for configuration space. EMAC uses only 8 KB of this memory and the remaining 8 KB is reserved for future use. Writing to the reserved memory may lead to unintended behavior.

72.18.1 EMAC memory map

EMAC base address: 4048_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	MAC Configuration (MAC_Configuration)	32	RW	0000_8000h
4h	MAC Extended Configuration (MAC_Ext_Configuration)	32	RW	0000_0000h
8h	MAC Packet Filter (MAC_Packet_Filter)	32	RW	0000_0000h
Ch	MAC Watchdog Timeout (MAC_Watchdog_Timeout)	32	RW	0000_0000h
10h	MAC Hash Table First 32 Bits (MAC_Hash_Table_Reg0)	32	RW	0000_0000h
14h	MAC Hash Table Second 32 Bits (MAC_Hash_Table_Reg1)	32	RW	0000_0000h
50h	MAC VLAN Tag (MAC_VLAN_Tag)	32	RW	0000_0000h
50h	MAC VLAN Tag Control (MAC_VLAN_Tag_Ctrl)	32	RW	0000_0000h
54h	MAC VLAN Tag Data (MAC_VLAN_Tag_Data)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 0 (MAC_VLAN_Tag_Filter0)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 1 (MAC_VLAN_Tag_Filter1)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 2 (MAC_VLAN_Tag_Filter2)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 3 (MAC_VLAN_Tag_Filter3)	32	RW	0000_0000h
58h	MAC VLAN Hash Table (MAC_VLAN_Hash_Table)	32	RW	0000_0000h
60h	MAC VLAN Inclusion Or Replacement (MAC_VLAN_Incl)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 0 (MAC_VLAN_Incl0)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 1 (MAC_VLAN_Incl1)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 2 (MAC_VLAN_Incl2)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 3 (MAC_VLAN_Incl3)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 4 (MAC_VLAN_Incl4)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 5 (MAC_VLAN_Incl5)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 6 (MAC_VLAN_Incl6)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 7 (MAC_VLAN_Incl7)	32	RW	0000_0000h
64h	Inner VLAN Tag Inclusion Or Replacement (MAC_Inner_VLAN_Incl)	32	RW	0000_0000h
70h	MAC Q0 Tx Flow Control (MAC_Q0_Tx_Flow_Ctrl)	32	RW	0000_0000h
90h	MAC Receive Flow Control (MAC_Rx_Flow_Ctrl)	32	RW	0000_0000h
94h	MAC RxQ Control 4 (MAC_RxQ_Ctrl4)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A0h	MAC RxQ Control 0 (MAC_RxQ_Ctrl0)	32	RW	0000_0000h
A4h	Receive Queue Control 1 (MAC_RxQ_Ctrl1)	32	RW	0000_0000h
A8h	MAC RxQ Control 2 (MAC_RxQ_Ctrl2)	32	RW	0000_0000h
B0h	MAC Interrupt Status (MAC_Interrupt_Status)	32	R	0000_0000h
B4h	MAC Interrupt Enable (MAC_Interrupt_Enable)	32	RW	0000_0000h
B8h	MAC Rx Transmit Status (MAC_Rx_Tx_Status)	32	R	0000_0000h
110h	MAC Version (MAC_Version)	32	R	0000_1051h
114h	MAC Debug (MAC_Debug)	32	R	0000_0000h
11Ch	MAC Hardware Feature 0 (MAC_HW_Feature0)	32	R	0E09_5135h
120h	MAC Hardware Feature 1 (MAC_HW_Feature1)	32	R	2118_29A6h
124h	MAC Hardware Feature 2 (MAC_HW_Feature2)	32	R	0404_1041h
128h	MAC Hardware Feature 3 (MAC_HW_Feature3)	32	R	2C37_0E31h
140h	MAC DPP FSM Interrupt Status (MAC_DPP_FSM_Interrupt_Status)	32	R	0000_0000h
148h	MAC FSM Control (MAC_FSM_Control)	32	RW	0000_0000h
14Ch	MAC FSM ACT Timer (MAC_FSM_ACT_Timer)	32	RW	0000_0000h
150h	SCS_REG 1 (SCS_REG1)	32	RW	0000_0000h
200h	MAC MDIO Address (MAC_MDIO_Address)	32	RW	0000_0000h
204h	MAC MDIO Data (MAC_MDIO_Data)	32	RW	0000_0000h
230h	MAC CSR Software Control (MAC_CSR_SW_Ctrl)	32	RW	0000_0000h
234h	MAC FPE Control STS (MAC_FPE_CTRL_STS)	32	RW	0000_0000h
240h	MAC Presentation Time (MAC_Presn_Time_ns)	32	R	0000_0000h
244h	MAC Presentation Time Update (MAC_Presn_Time_Updt)	32	RW	0000_0000h
300h	MAC Address 0 High (MAC_Address0_High)	32	RW	8000_FFFFh
304h	MAC Address 0 Low (MAC_Address0_Low)	32	RW	FFFF_FFFFh
308h	MAC Address 1 High (MAC_Address1_High)	32	RW	0000_FFFFh
30Ch	MAC Address 1 Low (MAC_Address1_Low)	32	RW	FFFF_FFFFh
310h	MAC Address 2 High (MAC_Address2_High)	32	RW	0000_FFFFh
314h	MAC Address 2 Low (MAC_Address2_Low)	32	RW	FFFF_FFFFh
700h	MMC Control (MMC_Control)	32	RW	0000_0000h
704h	MMC Receive Interrupt (MMC_Rx_Interrupt)	32	R	0000_0000h
708h	MMC Transmit Interrupt (MMC_Tx_Interrupt)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
70Ch	MMC Receive Interrupt Mask (MMC_Rx_Interrupt_Mask)	32	RW	0000_0000h
710h	MMC Transmit Interrupt Mask (MMC_Tx_Interrupt_Mask)	32	RW	0000_0000h
714h	Transmit Octet Count Good Bad (Tx_Octet_Count_Good_Bad)	32	R	0000_0000h
718h	Transmit Packet Count Good Bad (Tx_Packet_Count_Good_Bad)	32	R	0000_0000h
71Ch	Transmit Broadcast Packets Good (Tx_Broadcast_Packets_Good)	32	R	0000_0000h
720h	Transmit Multicast Packets Good (Tx_Multicast_Packets_Good)	32	R	0000_0000h
724h	Transmit 64-Octet Packets Good Bad (Tx_64Octets_Packets_Good_Bad)	32	R	0000_0000h
728h	Transmit 65 To 127 Octet Packets Good Bad (Tx_65To127Octets_Packets_Good_Bad)	32	R	0000_0000h
72Ch	Transmit 128 To 255 Octet Packets Good Bad (Tx_128To255Octets_Packets_Good_Bad)	32	R	0000_0000h
730h	Transmit 256 To 511 Octet Packets Good Bad (Tx_256To511Octets_Packets_Good_Bad)	32	R	0000_0000h
734h	Transmit 512 To 1023 Octet Packets Good Bad (Tx_512To1023Octets_Packets_Good_Bad)	32	R	0000_0000h
738h	Transmit 1024 To Max Octet Packets Good Bad (Tx_1024ToMaxOctets_Packets_Good_Bad)	32	R	0000_0000h
73Ch	Transmit Unicast Packets Good Bad (Tx_Unicast_Packets_Good_Bad)	32	R	0000_0000h
740h	Transmit Multicast Packets Good Bad (Tx_Multicast_Packets_Good_Bad)	32	R	0000_0000h
744h	Transmit Broadcast Packets Good Bad (Tx_Broadcast_Packets_Good_Bad)	32	R	0000_0000h
748h	Transmit Underflow Error Packets (Tx_Underflow_Error_Packets)	32	R	0000_0000h
74Ch	Transmit Single Collision Good Packets (Tx_Single_Collision_Good_Packets)	32	R	0000_0000h
750h	Transmit Multiple Collision Good Packets (Tx_Multiple_Collision_Good_Packets)	32	R	0000_0000h
754h	Transmit Deferred Packets (Tx_Deferred_Packets)	32	R	0000_0000h
758h	Transmit Late Collision Packets (Tx_Late_Collision_Packets)	32	R	0000_0000h
75Ch	Transmit Excessive Collision Packets (Tx_Excessive_Collision_Packets)	32	R	0000_0000h
760h	Transmit Carrier Error Packets (Tx_Carrier_Error_Packets)	32	R	0000_0000h
764h	Transmit Octet Count Good (Tx_Octet_Count_Good)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
768h	Transmit Packet Count Good (Tx_Packet_Count_Good)	32	R	0000_0000h
76Ch	Transmit Excessive Deferral Error (Tx_Excessive_Deferral_Error)	32	R	0000_0000h
770h	Transmit Pause Packets (Tx_Pause_Packets)	32	R	0000_0000h
774h	Transmit VLAN Packets Good (Tx_VLAN_Packets_Good)	32	R	0000_0000h
778h	Transmit O Size Packets Good (Tx_OSize_Packets_Good)	32	R	0000_0000h
780h	Receive Packets Count Good Bad (Rx_Packets_Count_Good_Bad)	32	R	0000_0000h
784h	Receive Octet Count Good Bad (Rx_Octet_Count_Good_Bad)	32	R	0000_0000h
788h	Receive Octet Count Good (Rx_Octet_Count_Good)	32	R	0000_0000h
78Ch	Receive Broadcast Packets Good (Rx_Broadcast_Packets_Good)	32	R	0000_0000h
790h	Receive Multicast Packets Good (Rx_Multicast_Packets_Good)	32	R	0000_0000h
794h	Receive CRC Error Packets (Rx_CRC_Error_Packets)	32	R	0000_0000h
798h	Receive Alignment Error Packets (Rx_Alignment_Error_Packets)	32	R	0000_0000h
79Ch	Receive Runt Error Packets (Rx_Runt_Error_Packets)	32	R	0000_0000h
7A0h	Receive Jabber Error Packets (Rx_Jabber_Error_Packets)	32	R	0000_0000h
7A4h	Receive Undersize Packets Good (Rx_Undersize_Packets_Good)	32	R	0000_0000h
7A8h	Receive Oversize Packets Good (Rx_Oversize_Packets_Good)	32	R	0000_0000h
7ACh	Receive 64 Octets Packets Good Bad (Rx_64Octets_Packets_Good_Bad)	32	R	0000_0000h
7B0h	Receive 65-127 Octets Packets Good Bad (Rx_65To127Octets_Packets_Good_Bad)	32	R	0000_0000h
7B4h	Receive 128-255 Octets Packets Good Bad (Rx_128To255Octets_Packets_Good_Bad)	32	R	0000_0000h
7B8h	Receive 256-511 Octets Packets Good Bad (Rx_256To511Octets_Packets_Good_Bad)	32	R	0000_0000h
7BCh	Receive 512-1023 Octets Packets Good Bad (Rx_512To1023Octets_Packets_Good_Bad)	32	R	0000_0000h
7C0h	Receive 1024 To Max Octets Good Bad (Rx_1024ToMaxOctets_Packets_Good_Bad)	32	R	0000_0000h
7C4h	Receive Unicast Packets Good (Rx_Unicast_Packets_Good)	32	R	0000_0000h
7C8h	Receive Length Error Packets (Rx_Length_Error_Packets)	32	R	0000_0000h
7CCh	Receive Out of Range Type Packet (Rx_Out_Of_Range_Type_Packets)	32	R	0000_0000h
7D0h	Receive Pause Packets (Rx_Pause_Packets)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
7D4h	Receive FIFO Overflow Packets (Rx_FIFO_Overflow_Packets)	32	R	0000_0000h
7D8h	Receive VLAN Packets Good Bad (Rx_VLAN_Packets_Good_Bad)	32	R	0000_0000h
7DCh	Receive Watchdog Error Packets (Rx_Watchdog_Error_Packets)	32	R	0000_0000h
7E0h	Receive Receive Error Packets (Rx_Receive_Error_Packets)	32	R	0000_0000h
7E4h	Receive Control Packets Good (Rx_Control_Packets_Good)	32	R	0000_0000h
8A0h	MMC Transmit FPE Fragment Counter Interrupt Status (MMC_FPE_Tx_Interrupt)	32	R	0000_0000h
8A4h	MMC FPE Transmit Interrupt Mask (MMC_FPE_Tx_Interrupt_Mask)	32	RW	0000_0000h
8A8h	Transmit FPE Fragment Counter (MMC_Tx_FPE_Fragment_Cntr)	32	R	0000_0000h
8ACh	Transmit Hold Request Counter (MMC_Tx_Hold_Req_Cntr)	32	R	0000_0000h
8C0h	MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt)	32	R	0000_0000h
8C4h	MMC FPE Receive Interrupt Mask (MMC_FPE_Rx_Interrupt_Mask)	32	RW	0000_0000h
8C8h	MMC Receive Packet Assembly Error Counter (MMC_Rx_Packet_Assembly_Err_Cntr)	32	R	0000_0000h
8CCh	MMC Receive Packet SMD Error Counter (MMC_Rx_Packet_SMD_Err_Cntr)	32	R	0000_0000h
8D0h	MMC Receive Packet Assembly OK Counter (MMC_Rx_Packet_Assembly_OK_Cntr)	32	R	0000_0000h
8D4h	MMC Receive FPE Fragment Counter (MMC_Rx_FPE_Fragment_Cntr)	32	R	0000_0000h
900h	MAC Layer 3 Layer 4 Control 0 (MAC_L3_L4_Control0)	32	RW	0000_0000h
904h	MAC Layer 4 Address 0 (MAC_Layer4_Address0)	32	RW	0000_0000h
910h	MAC Layer 3 Address 0 Reg 0 (MAC_Layer3_Addr0_Reg0)	32	RW	0000_0000h
914h	MAC Layer 3 Address 1 Reg 0 (MAC_Layer3_Addr1_Reg0)	32	RW	0000_0000h
918h	MAC Layer 3 Address 2 Reg 0 (MAC_Layer3_Addr2_Reg0)	32	RW	0000_0000h
91Ch	MAC Layer 3 Address 3 Reg 0 (MAC_Layer3_Addr3_Reg0)	32	RW	0000_0000h
930h	MAC L3 L4 Control 1 (MAC_L3_L4_Control1)	32	RW	0000_0000h
934h	MAC Layer 4 Address 1 (MAC_Layer4_Address1)	32	RW	0000_0000h
940h	MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1)	32	RW	0000_0000h
944h	MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1)	32	RW	0000_0000h
948h	MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1)	32	RW	0000_0000h
94Ch	MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
960h	MAC L3 L4 Control 2 (MAC_L3_L4_Control2)	32	RW	0000_0000h
964h	MAC Layer 4 Address 2 (MAC_Layer4_Address2)	32	RW	0000_0000h
970h	MAC Layer 3 Address 0 Reg 2 (MAC_Layer3_Addr0_Reg2)	32	RW	0000_0000h
974h	MAC Layer 3 Address 1 Reg 2 (MAC_Layer3_Addr1_Reg2)	32	RW	0000_0000h
978h	MAC Layer 3 Address 2 Reg 2 (MAC_Layer3_Addr2_Reg2)	32	RW	0000_0000h
97Ch	MAC Layer 3 Address 3 Reg 2 (MAC_Layer3_Addr3_Reg2)	32	RW	0000_0000h
990h	MAC L3 L4 Control 3 (MAC_L3_L4_Control3)	32	RW	0000_0000h
994h	MAC Layer 4 Address 3 (MAC_Layer4_Address3)	32	RW	0000_0000h
9A0h	MAC Layer 3 Address 0 Reg 3 (MAC_Layer3_Addr0_Reg3)	32	RW	0000_0000h
9A4h	MAC Layer 3 Address 1 Reg 3 (MAC_Layer3_Addr1_Reg3)	32	RW	0000_0000h
9A8h	MAC Layer 3 Address 2 Reg 3 (MAC_Layer3_Addr2_Reg3)	32	RW	0000_0000h
9ACh	MAC Layer 3 Address 3 Reg 3 (MAC_Layer3_Addr3_Reg3)	32	RW	0000_0000h
B00h	MAC Timestamp Control (MAC_Timestamp_Control)	32	RW	0000_2000h
B04h	MAC Sub Second Increment (MAC_Sub_Second_Increment)	32	RW	0000_0000h
B08h	MAC System Time In Seconds (MAC_System_Time_Seconds)	32	R	0000_0000h
B0Ch	MAC System Time In Nanoseconds (MAC_System_Time_Nanoseconds)	32	R	0000_0000h
B10h	MAC System Time Seconds Update (MAC_System_Time_Seconds_Update)	32	RW	0000_0000h
B14h	MAC System Time Nanoseconds Update (MAC_System_Time_Nanoseconds_Update)	32	RW	0000_0000h
B18h	MAC Timestamp Addend (MAC_Timestamp_Addend)	32	RW	0000_0000h
B1Ch	MAC System Time Higher Word In Seconds (MAC_System_Time_Higher_Word_Seconds)	32	RW	0000_0000h
B20h	MAC Timestamp Status (MAC_Timestamp_Status)	32	R	0000_0000h
B30h	MAC Transmit Timestamp Status In Nanoseconds (MAC_Tx_Timestamp_Status_Nanoseconds)	32	R	0000_0000h
B34h	MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds)	32	R	0000_0000h
B50h	MAC Timestamp Ingress Asymmetry Correction (MAC_Timestamp_Ingress_Asym_Corr)	32	RW	0000_0000h
B54h	MAC Timestamp Egress Asymmetry Correction (MAC_Timestamp_Egress_Asym_Corr)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B58h	MAC Timestamp Ingress Correction In Nanoseconds (MAC_Timestamp_Ingress_Corr_Nanosecond)	32	RW	0000_0000h
B5Ch	MAC Timestamp Egress Correction In Nanoseconds (MAC_Timestamp_Egress_Corr_Nanosecond)	32	RW	0000_0000h
B60h	MAC Timestamp Ingress Correction In Subnanoseconds (MAC_Timestamp_Ingress_Corr_Subnanosec)	32	RW	0000_0000h
B64h	MAC Timestamp Engress Correction In Subnanoseconds (MAC_Timestamp_Egress_Corr_Subnanosec)	32	RW	0000_0000h
B68h	MAC Timestamp Ingress Latency (MAC_Timestamp_Ingress_Latency)	32	R	0000_0000h
B6Ch	MAC Timestamp Egress Latecy (MAC_Timestamp_Egress_Latency)	32	R	0000_0000h
B70h	MAC PPS Control (MAC_PPS_Control)	32	RW	0000_0000h
B80h	MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds)	32	RW	0000_0000h
B84h	MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds)	32	RW	0000_0000h
B88h	MAC PPS0 Interval (MAC_PPS0_Interval)	32	RW	0000_0000h
B8Ch	MAC PPS0 Width (MAC_PPS0_Width)	32	RW	0000_0000h
B90h	MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds)	32	RW	0000_0000h
B94h	MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds)	32	RW	0000_0000h
B98h	MAC PPS1 Interval (MAC_PPS1_Interval)	32	RW	0000_0000h
B9Ch	MAC PPS1 Width (MAC_PPS1_Width)	32	RW	0000_0000h
BA0h	MAC PPS2 Taget Time In Seconds (MAC_PPS2_Target_Time_Seconds)	32	RW	0000_0000h
BA4h	MAC PPS2 Target Time In Nanoseconds (MAC_PPS2_Target_Time_Nanoseconds)	32	RW	0000_0000h
BA8h	MAC PPS2 Interval (MAC_PPS2_Interval)	32	RW	0000_0000h
BACh	MAC PPS2 Width (MAC_PPS2_Width)	32	RW	0000_0000h
BB0h	MAC PPS3 Target Time In Seconds (MAC_PPS3_Target_Time_Seconds)	32	RW	0000_0000h
BB4h	MAC PPS3 Target Time In Nanoseconds (MAC_PPS3_Target_Time_Nanoseconds)	32	RW	0000_0000h
BB8h	MAC PPS3 Interval (MAC_PPS3_Interval)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
BBCh	MAC PPS3 Width (MAC_PPS3_Width)	32	RW	0000_0000h
C00h	MTL Operation Mode (MTL_Operation_Mode)	32	RW	0000_0000h
C08h	MTL Debug Control (MTL_DBG_CTL)	32	RW	0000_0000h
C0Ch	MTL Debug Status (MTL_DBG_STS)	32	RW	0000_0018h
C10h	MTL FIFO Debug Data (MTL_FIFO_Debug_Data)	32	RW	0000_0000h
C20h	MTL Interrupt Status (MTL_Interrupt_Status)	32	R	0000_0000h
C30h	MTL Receive Queue DMA Map 0 (MTL_RxQ_DMA_Map0)	32	RW	0000_0000h
C40h	MTL TBS Control (MTL_TBS_CTRL)	32	RW	0000_0000h
C50h	MTL EST Control (MTL_EST_Control)	32	RW	0000_0000h
C58h	MTL EST Status (MTL_EST_Status)	32	RW	0000_0000h
C60h	MTL EST Scheduling Error (MTL_EST_Sch_Error)	32	RW	0000_0000h
C64h	MTL EST Frame Size Error (MTL_EST_Frm_Size_Error)	32	RW	0000_0000h
C68h	MTL EST Frame Size Capture (MTL_EST_Frm_Size_Capture)	32	R	0000_0000h
C70h	MTL EST Interrupt Enable (MTL_EST_Intr_Enable)	32	RW	0000_0000h
C80h	MTL EST GCL Control (MTL_EST_GCL_Control)	32	RW	0000_0000h
C84h	MTL EST GCL Data (MTL_EST_GCL_Data)	32	RW	0000_0000h
C90h	MTL FPE Control Status (MTL_FPE_CTRL_STS)	32	RW	0000_0000h
C94h	MTL FPE Advance (MTL_FPE_Advance)	32	RW	0000_0000h
CA0h	MTL Rx Parser Control Status (MTL_RXP_Control_Status)	32	RW	803F_003Fh
CA4h	MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status)	32	RW	0000_0000h
CA8h	MTL Rx Parser Drop Count (MTL_RXP_Drop_Cnt)	32	R	0000_0000h
CACh	MTL Rx Parser Error Count (MTL_RXP_Error_Cnt)	32	R	0000_0000h
CB0h	MTL Rx Parser Indirect Access Control Status (MTL_RXP_Indirect_Acc_Control_Status)	32	RW	0000_0000h
CB4h	MTL Rx Parser Indirect Access Data (MTL_RXP_Indirect_Acc_Data)	32	R	0000_0000h
CC0h	MTL ECC Control (MTL_ECC_Control)	32	RW	0000_000Fh
CC4h	MTL Safety Interript Status (MTL_Safety_Interrupt_Status)	32	R	0000_0000h
CC8h	MTL ECC Interrupt Enable (MTL_ECC_Interrupt_Enable)	32	RW	0000_1111h
CCCh	MTL ECC Interrupt Status (MTL_ECC_Interrupt_Status)	32	RW	0000_0000h
CD0h	MTL ECC Error Status (MTL_ECC_Err_Sts_Rctl)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
CD4h	MTL ECC Error Adress Status (MTL_ECC_Err_Addr_Status)	32	R	0000_0000h
CD8h	MTL ECC Error Control Status (MTL_ECC_Err_Cntr_Status)	32	R	0000_0000h
CE0h	MTL DPP Control (MTL_DPP_Control)	32	RW	0000_0003h
D00h	MTL Tx Queue 0 Operation Mode (MTL_TxQ0_Operation_Mode)	32	RW	0000_0000h
D04h	MTL Tx Queue 0 Underflow (MTL_TxQ0_Underflow)	32	R	0000_0000h
D08h	MTL Tx Queue 0 Debug (MTL_TxQ0_Debug)	32	R	0000_0000h
D14h	MTL Tx Queue 0 ETS Status (MTL_TxQ0_ETS_Status)	32	R	0000_0000h
D18h	MTL Tx Queue Quantum Weight (MTL_TxQ0_Quantum_Weight)	32	RW	0000_0000h
D2Ch	MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status)	32	RW	0000_0000h
D30h	MTL Rx Queue 0 Operation Mode (MTL_RxQ0_Operation_Mode)	32	RW	0000_0000h
D34h	MTL Rx Queue Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt)	32	R	0000_0000h
D38h	MTL Rx Queue 0 Debug (MTL_RxQ0_Debug)	32	R	0000_0000h
D3Ch	MTL Rx Queue 0 Control 0 (MTL_RxQ0_Control)	32	RW	0000_0000h
D40h	MTL Tx Queue 1 Operation Mode (MTL_TxQ1_Operation_Mode)	32	RW	0000_0000h
D44h	MTL Tx Queue 1 Underflow (MTL_TxQ1_Underflow)	32	R	0000_0000h
D48h	MTL Tx Queue 1 Debug (MTL_TxQ1_Debug)	32	R	0000_0000h
D50h	MTL Tx Queue 1 ETS Control (MTL_TxQ1_ETS_Control)	32	RW	0000_0000h
D54h	MTL Tx Queue 1 ETS Status (MTL_TxQ1_ETS_Status)	32	R	0000_0000h
D58h	MTL Tx Queue 1 Quantum Weight (MTL_TxQ1_Quantum_Weight)	32	RW	0000_0000h
D5Ch	MTL Tx Queue 1 Sendslope Credit (MTL_TxQ1_SendSlopeCredit)	32	RW	0000_0000h
D60h	MTL Tx Queue 1 HiCredit (MTL_TxQ1_HiCredit)	32	RW	0000_0000h
D64h	MTL Tx Queue 1 LoCredit (MTL_TxQ1_LoCredit)	32	RW	0000_0000h
D6Ch	MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status)	32	RW	0000_0000h
D70h	MTL Rx Queue 1 Operation Mode (MTL_RxQ1_Operation_Mode)	32	RW	0000_0000h
D74h	MTL Rx Queue 1 Missed Packet Overflow Counter (MTL_RxQ1_Missed_Packet_Overflow_Cnt)	32	R	0000_0000h
D78h	MTL Rx Queue 1 Debug (MTL_RxQ1_Debug)	32	R	0000_0000h
D7Ch	MTL Rx Queue 1 Control (MTL_RxQ1_Control)	32	RW	0000_0000h
1000h	DMA Mode (DMA_Mode)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1004h	DMA System Bus Mode (DMA_SysBus_Mode)	32	RW	0000_0000h
1008h	DMA Interrupt Status (DMA_Interrupt_Status)	32	R	0000_0000h
100Ch	DMA Debug Status 0 (DMA_Debug_Status0)	32	R	0000_0000h
1050h	DMA TBS Control (DMA_TBS_CTRL)	32	RW	0000_0000h
1080h	DMA Safety Interrupt Status (DMA_Safety_Interrupt_Status)	32	R	0000_0000h
1100h	DMA Channel 0 Control (DMA_CH0_Control)	32	RW	0000_0000h
1104h	DMA Channel Tx Control (DMA_CH0_Tx_Control)	32	RW	0000_0000h
1108h	DMA Channel Rx Control (DMA_CH0_Rx_Control)	32	RW	0000_0000h
1114h	DMA Channel 0 Tx Descriptor List Address (DMA_CH0_TxDesc_List_Address)	32	RW	0000_0000h
111Ch	DMA Channel 0 Rx Descriptor List Address (DMA_CH0_RxDesc_List_Address)	32	RW	0000_0000h
1120h	DMA Channel 0 Tx Descriptor Tail Pointer (DMA_CH0_TxDesc_Tail_Pointer)	32	RW	0000_0000h
1128h	DMA Channeli 0 Rx Descriptor List Pointer (DMA_CH0_RxDesc_Tail_Pointer)	32	RW	0000_0000h
112Ch	DMA Channel 0 Tx Descriptor Ring Length (DMA_CH0_TxDesc_Ring_Length)	32	RW	0000_0000h
1130h	DMA Channel 0 Rx Descriptor Ring Length (DMA_CH0_RxDesc_Ring_Length)	32	RW	0000_0000h
1134h	DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)	32	RW	0000_0000h
1138h	DMA Channel 0 Rx Interrupt Watchdog Timer (DMA_CH0_Rx_Interrupt_Watchdog_Timer)	32	RW	0000_0000h
113Ch	DMA Channel 0 Slot Function Control Status (DMA_CH0_Slot_Function_Control_Status)	32	RW	0000_07C0h
1144h	DMA Channel 0 Current Application Transmit Descriptor (DMA_CH0_Current_App_TxDesc)	32	R	0000_0000h
114Ch	DMA Channel 0 Current Application Receive Descriptor (DMA_CH0_Current_App_RxDesc)	32	R	0000_0000h
1154h	DMA Channel 0 Current Application Transmit Descriptor (DMA_CH0_Current_App_TxBuffer)	32	R	0000_0000h
115Ch	DMA Channel 0 Current Application Receive Buffer (DMA_CH0_Current_App_RxBuffer)	32	R	0000_0000h
1160h	DMA Channel 0 Status (DMA_CH0_Status)	32	RW	0000_0000h
1164h	DMA Channel 0 Miss Frame Counter (DMA_CH0_Miss_Frame_Cnt)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1168h	DMA Channel 0 Rx Parser Accept Count (DMA_CH0_RXP_Accept_Cnt)	32	R	0000_0000h
116Ch	DMA Channel 0 Rx ERI Count (DMA_CH0_RX_ERI_Cnt)	32	R	0000_0000h
1180h	DMA Channel 1 Control (DMA_CH1_Control)	32	RW	0000_0000h
1184h	DMA Channel 1 Tx Control (DMA_CH1_Tx_Control)	32	RW	0000_0000h
1188h	DMA Channel 1 Rx Control (DMA_CH1_Rx_Control)	32	RW	0000_0000h
1194h	DMA Channel 1 Tx Descriptor List Address (DMA_CH1_TxDesc_List_Address)	32	RW	0000_0000h
119Ch	DMA Channel 1 Rx Descriptor List Address (DMA_CH1_RxDesc_List_Address)	32	RW	0000_0000h
11A0h	DMA Channel 1 Tx Descriptor Tail Pointer (DMA_CH1_TxDesc_Tail_Pointer)	32	RW	0000_0000h
11A8h	DMA Channel 1 Rx Descriptor Tail Pointer (DMA_CH1_RxDesc_Tail_Pointer)	32	RW	0000_0000h
11ACh	DMA Channel 1 Tx Descriptor Ring Length (DMA_CH1_TxDesc_Ring_Length)	32	RW	0000_0000h
11B0h	DMA Channel 1 Rx Descriptor Ring Length (DMA_CH1_RxDesc_Ring_Length)	32	RW	0000_0000h
11B4h	DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)	32	RW	0000_0000h
11B8h	DMA Channel 1 Rx Interrupt Watchdog Timer (DMA_CH1_Rx_Interrupt_Watchdog_Timer)	32	RW	0000_0000h
11BCh	DMA Channel 1 Slot Function Control Status (DMA_CH1_Slot_Function_Control_Status)	32	RW	0000_07C0h
11C4h	DMA Channel 1 Current Application Transmit Descriptor (DMA_CH1_Current_App_TxDesc)	32	R	0000_0000h
11CCh	DMA Channel 1 Current Application Receive Descriptor (DMA_CH1_Current_App_RxDesc)	32	R	0000_0000h
11D4h	DMA Channel 1 Current Application Transmit Buffer (DMA_CH1_Current_App_TxBuffer)	32	R	0000_0000h
11DCh	DMA Channel 1 Current Application Receive Buffer (DMA_CH1_Current_App_RxBuffer)	32	R	0000_0000h
11E0h	DMA Channel 1 Status (DMA_CH1_Status)	32	RW	0000_0000h
11E4h	DMA Channel 1 Miss Frame Counter (DMA_CH1_Miss_Frame_Cnt)	32	R	0000_0000h
11E8h	DMA Channel 1 Rx Parser Accept Count (DMA_CH1_RXP_Accept_Cnt)	32	R	0000_0000h
11ECh	DMA Channel 1 Rx ERI Count (DMA_CH1_RX_ERI_Cnt)	32	R	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value

72.18.2 MAC Configuration (MAC_Configuration)

Offset

Register	Offset
MAC_Configuration	0h

Function

Establishes MAC's operating mode.

Table 531. Packet length based on CST and ACS bits

Received packet length	CST	ACS	FCS stripping done
<1536	—	0	No
	—	1	Yes (for Ethernet packets)
>=1536	0	—	No
	1	—	Yes (for Type packets)
<1536	—	0	No
	—	1	Yes (for Ethernet packets)

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	SARC			IPC	IPG			GPSLCE	S2KP	CST	ACS	WD	Reserved	JD	JE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PS	FES	DM	LM	ECRSFD	DO	DCRS	DR	Reserved	BL		DC	PRELEN		TE	RE
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 SARC	Source Address Insertion Or Replacement Control Controls the source address insertion or replacement for all the transmitted packets. Bit 30 of this field specifies which MAC Address register (0 or 1) is used for source address insertion or replacement based on the values of bits [29:28]: • 2'b0x: The mti_sa_ctrl_i and ati_sa_ctrl_i input signals control the SA field generation. • 2'b10: If bit 30 = 0, MAC inserts the content of the MAC Address 0 registers in the SA field of all the transmitted packets. If bit 30 = 1 and the Enable MAC Address Register 1 option is selected while configuring the core, MAC inserts the content of the MAC Address 1 registers in the SA field of all the transmitted packets. • 2'b11: If bit 30 = 0, MAC replaces the content of the MAC Address 0 registers in the SA field of all the transmitted packets. If bit 30 = 1 and the MAC Address Register 1 is enabled, MAC replaces the content of the MAC Address 1 registers in the SA field of all the transmitted packets. **NOTE** Changes to this field take effect only on the start of a packet. If you write to the field when a packet is being transmitted, only the subsequent packet can use the updated value. This means that the current packet does not use the updated value. 000b - mti_sa_ctrl_i and ati_sa_ctrl_i input signals control the SA field generation 010b - Contents of are inserted in the SA field 011b - Contents of replace the SA field 110b - Contents of are inserted in the SA field 111b - Contents of replace the SA field
27 IPC	Checksum Offload Indicates the status of IP header or payload checksum checking. If this field is 1, the field enables IPv4 header checksum checking and IPv4 or IPv6 TCP, UDP, or ICMP payload checksum checking. When the field becomes 0, the COE function in the receiver is disabled. The Layer 3 and Layer 4 packet filter and enable split header features automatically select the IPC full checksum offload engine on the receive side. When any of these features are enabled, you must write 1 to this field. 0b - Disabled 1b - Enabled
26-24 IPG	Inter-Packet Gap Controls the minimum IPG between packets during transmission. The range of minimum IPG is valid in Full-Duplex mode, and in this mode, the minimum IPG can be configured only for 64-bit times (IPG = 100). Lower values are not considered.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When a JAM pattern is being transmitted because of backpressure activation, MAC does not consider the minimum IPG. This function (IPG less than 96-bit times) is valid only when MAC_Ext_Configuration[EIPGEN] is 0. If this field is 1, the minimum IPG (greater than 96-bit times) is controlled according to the field's description. 000b - 96-bit times IPG 001b - 88-bit times IPG 010b - 80-bit times IPG 011b - 72-bit times IPG 100b - 64-bit times IPG 101b - 56-bit times IPG 110b - 48-bit times IPG 111b - 40-bit times IPG
23 GPSLCE	Giant Packet Size Limit Control Enable Enables and disables giant packet size limit control. If this field = 1, MAC considers the value of MAC_Ext_Configuration[GPSL] to declare a received packet as a giant packet. This field must be programmed to have a size of more than 1,518 bytes. Otherwise, MAC considers 1,518 bytes as the giant packet limit. When this field becomes 0, MAC considers the received packet as a giant packet if its size is greater than 1,518 bytes (1522 bytes for a tagged packet). The WD, JE, and S2KP fields have a higher precedence over this field, which means MAC considers a received packet as a giant packet when its size is greater than 9,018 bytes (9,022 bytes for a tagged packet) with JE = 1 and the size of the jumbo packet greater than 2,000 bytes and S2KP = 1. If the WD field is 1, the field terminates the received packet if it reaches the watchdog limit. Therefore, the programmed giant packet limit must be less than the watchdog limit to get the giant packet status. 0b - Disabled 1b - Enabled
22 S2KP	IEEE 802.3 Support For 2K Packets Indicates the status of IEEE 802.3 support for 2K packets. If this field is 1, MAC considers all the packets with up to 2,000-byte length as normal. When the JE field is not 1, MAC considers all the received packets of size more than 2K bytes as giant packets. When this field becomes 0 and the JE field is not 1, MAC considers all the received packets of size more than 1,518 bytes (1,522 bytes for tagged) as giant packets. For more information about how the setting of this field and the JE field impacts the giant packet status, see this status based on the S2KP and JE fields. If the JE field is 1, writing 1 to the S2KP field has no effect on the giant packet status. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 CST	CRC Stripping For Type Packets Indicates the status of CRC stripping for Type packets. If this field is 1, the last four bytes (FCS) of all the Ether-type packets are stripped and dropped before forwarding the packet to the application. This field is valid only when the receive packet size is more than or equal to 1536 bytes. For information about how the settings of this field and the ACS field impact the packet length, see MAC Configuration (MAC_Configuration). 0b - Disabled 1b - Enabled
20 ACS	Automatic Pad Or CRC Stripping Indicates the status of automatic pad or CRC stripping. If this field is 1, MAC strips the pad or the FCS field on the incoming packets only if the size of the packets is less than 1,536 bytes. All the received packets with a size greater than or equal to 1,536 bytes are passed to the application without stripping the pad or the FCS field. When this field becomes 0, MAC passes all the incoming packets to the application, without any modification. For information about how the settings of this field and the CST field impact the packet length, see MAC Configuration (MAC_Configuration). 0b - Disabled 1b - Enabled
19 WD	Watchdog Disable Indicates the status of watchdog. If this field is 1, MAC disables the watchdog timer on the receiver. MAC can receive packets of up to 16,383 bytes. When this field becomes 0, MAC does not allow more than 2,048 bytes (10,240 if the JE field = 1) of the packet being received. MAC cuts off bytes received after 2,048. 0b - Enabled 1b - Disabled
18 —	Reserved
17 JD	Jabber Disable Indicates the status of jabber. If this field is 1, MAC disables the jabber timer on the transmitter. It can transfer packets of up to 16,383 bytes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 0 and the application sends more than 2,048 bytes of data (10,240 if the JE field = 1) during transmission, MAC does not send the rest of the bytes in that packet. 0b - Enabled 1b - Disabled
16 JE	Jumbo Packet Enable Indicates the status of jumbo packets. If this field is 1, MAC allows jumbo packets of 9,018 bytes (9,022 bytes for VLAN-tagged packets) without reporting a giant packet error in the receive packet status. 0b - Disabled 1b - Enabled
15 PS	Port Select Selects the Ethernet line speed. This field, along with the FES field, selects the exact line speed. In the 10/100 Mbit/s-only (always 1) or 1000 Mbit/s-only (always 0) configurations, this field is read-only (RO) with an appropriate value. By default, with 10/100/1000 Mbit/s configurations, this field is read-write (R/W). The mac_speed_o[1] signal reflects the value of this field. 0b - For 1000 or 2500 Mbit/s operations 1b - For 10 or 100 Mbit/s operations
14 FES	Speed Indicates the speed. This field selects the speed mode. The mac_speed_o[0] signal indicates the value of this field. 0b - 10 Mbit/s if PS = 1 and 1 Gbps if PS = 0 1b - 100 Mbit/s if PS = 1 and 2.5 Gbps if PS = 0
13 DM	Duplex Mode Indicates the mode in which MAC operates. If this field is 1, MAC operates in Full-Duplex mode in which it can transmit and receive simultaneously. The field can only be read, with 1 as its default value, in full-duplex-only configurations. 0b - Half-duplex mode 1b - Full-duplex mode
12 LM	Loopback Mode Indicates the status of Loopback mode If this field is 1, MAC operates in Loopback mode at GMII or MII. This mode requires the (G)MII Rx clock input (clk_rx_i) signal to function properly because the transmit clock is not internally looped back.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
11 ECRSFD	Enable Carrier Sense In Full-Duplex Mode Indicates whether ECRSFD is enabled or disabled. If this field is 1, the MAC transmitter checks the CRS signal before packet transmission in Full-Duplex mode. MAC starts transmitting only when the CRS signal is low. If this field becomes 0, the MAC transmitter ignores the status of the CRS signal. 0b - Disabled 1b - Enabled
10 DO	Disable Receive Own Enables or disables receive own. If this field is 1, MAC disables the reception of packets when the gmii_txen_o signal asserts in Half-Duplex mode. When the field is 0, MAC receives all the packets that PHY sent. This field is not applicable in Full-Duplex mode. 0b - Enabled 1b - Disabled
9 DCRS	Disable Carrier Sense During Transmission Enables or disables carrier sense during transmission. If this field is 1, the MAC transmitter ignores the (G)MII CRS signal during packet transmission in Half-Duplex mode. As a result, no errors generate because of loss of carrier or no carrier during transmission. When this field becomes 0, the MAC transmitter generates errors because of carrier sense. MAC can even abort the transmission. 0b - Enabled 1b - Disabled
8 DR	Disable Retry Enables or disables retry attempts. If this field is 1, MAC attempts only one transmission. When a collision occurs on GMII or MII, MAC ignores the current packet transmission and reports a packet abort with excessive collision errors in the transmit packet status. If this field becomes 0, MAC retries based on the settings of the BL field of this register. The settings of this field apply only in Half-Duplex mode. 0b - Enabled 1b - Disabled
7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6-5 BL	Back-Off Limit Determines the random integer number (r) of slot time delays (4,096 bit times for 1000/2500 Mbit/s; 512 bit times for 10/100 Mbit/s) for which MAC waits before rescheduling a transmission attempt during retry attempts after a collision. • *n* = retransmission attempt • *r* = random integer that takes the value in the range 0 <= r < 2^k This field is applicable only in Half-Duplex mode. 00b - k = min(n,10) 01b - k = min(n,8) 10b - k = min(n,4) 11b - k = min(n,1)
4 DC	Deferral Check Indicates the status of the deferral check function. If this field is 1, the deferral check function is enabled in MAC, which issues a packet abort status, with Tx_Excessive_Deferral_Error[TXEXSDEF] = 1 in the transmit packet status, when the transmit state machine is deferred for more than 24,288 bit times in 10 Mbit/s or 100 Mbit/s mode. If MAC is configured for a 1000/2500 Mbit/s operation, the threshold for deferral is 155,680 bit times. Deferral begins when the transmitter is ready to transmit, but it is prevented because of an active carrier sense signal (CRS) on GMII or MII. The defer time is not cumulative. For example, if the transmitter defers for 10,000 bit times because the CRS signal is active, and then the CRS signal becomes inactive, the transmitter transmits and collision happens. Because of this collision, the transmitter needs to back off and then defer again after back off completion. In such a scenario, the deferral timer resets to 0 and restarts. When this field becomes 0, the deferral check function disables and MAC defers until the CRS signal becomes inactive. This field is applicable only in Half-Duplex mode. 0b - Disabled 1b - Enabled
3-2 PRELEN	Preamble Length for Transmit Packets Controls the number of preamble bytes that are added to the beginning of every transmit packet. Preamble reduction occurs only when MAC operates in Full-Duplex mode. 00b - 7 bytes 01b - 5 bytes 10b - 3 bytes 11b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 TE	Transmitter Enable Indicates the status of the transmitter. • If this field is 1, the MAC transmit state machine is enabled for transmission on GMII or MII. • When the field becomes 0, the MAC transmit state machine is disabled after it completes the transmission of the current packet. The transmit state machine does not transmit any more packets. 0b - Disabled 1b - Enabled
0 RE	Receiver Enable Indicates the status of the receiver. If this field is 1, MAC's receive state machine is enabled for receiving packets from GMII or MII. When this field becomes 0, the MAC receive state machine is disabled after it completes the reception of the current packet. The receive state machine does not receive any more packets from GMII or MII. 0b - Disabled 1b - Enabled

72.18.3 MAC Extended Configuration (MAC_Ext_Configuration)

Offset

Register	Offset
MAC_Ext_Configuration	4h

Function

Establishes MAC's operating mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	Reserved	EIPG					EIPGEN	Reserved	Reserved			PDC	USP	SPEN	DCRCC
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29-25 EIPG	Extended Inter-Packet Gap Indicates the value of the extended inter-packet gap. The value of this field is applicable when EIPGEN = 1. The most-significant bits of this field, along with those of MAC_Configuration[IPG], render the minimum IPG greater than 96-bit times in steps of 8-bit times. 8'h00: 104-bit times 8'h01: 112-bit times 8'h02: 120-bit times ---------- 8'hFF: 2144-bit times
24 EIPGEN	Extended Inter-Packet Gap Enable Indicates the status of the extended inter-packet gap. • If this field is 1, MAC interprets the EIPG field of this register and the IPG field of MAC Configuration (MAC_Configuration) together as minimum IPG greater than 96-bit times in steps of 8-bit times. • If this field is 0, MAC ignores the EIPG field of this regsiter and interprets the IPG field of MAC Configuration (MAC_Configuration) as minimum IPG less than or equal to 96-bit times in steps of 8-bit times. **NOTE** You must enable the extended inter-packet gap feature when operating in Full-Duplex mode only. There could be undesirable effects on the back-pressure function and frame transmission if the feature is enabled in Half-Duplex mode. 0b - Disabled 1b - Enabled
23 —	Reserved
22-20 —	Reserved
19 PDC	Packet Duplication Control Indicates the packet duplication control status.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, the received packet with multicast or broadcast destination address is routed to multiple receive DMA channels. MAC_Address0_High[DCS] identifies the receive DMA channels (corresponding to the MAC Address register) that match the multicast or broadcast destination addresses in the received packet. The DCS field is interpreted to be a one-hot value, each bit corresponding to the receive DMA channel. • If this field is 0, the received packet routes to a single receive DMA channel. MAC_Address0_High[DCS] identifies the receive DMA channel corresponding to the MAC Address register that matches the destination address in the received packet. The DCS field is interpreted as a binary value. 0b - Disabled 1b - Enabled
18 USP	Unicast Slow Protocol Packet Detect Indicates the status of unicast slow-protocol packet detection. • If this field is 1, MAC detects the slow-protocol packets with unicast address of the station specified in MAC Address 0 High (MAC_Address0_High) and MAC Address 0 Low (MAC_Address0_Low). MAC also detects the slow-protocol packets with the slow-protocol multicast address (01-80-C2-00-00-02). • If this field is 0, MAC detects only slow-protocol packets with the slow protocol multicast address specified in IEEE 802.3-2015, section 5. 0b - Disabled 1b - Enabled
17 SPEN	Slow Protocol Detection Enable Enables or disables slow protocol detection. • If this field is 0, EMAC forwards all error-free, slow-protocol packets to the application. EMAC considers such packets as normal-type packets. • If this field is 1, EMAC processes the slow-protocol packets (Ether type 8809h) and provides the slow protocol sub-type and code fields in receive status. 0b - Disabled 1b - Enabled
16 DCRCC	Disable CRC Checking For Received Packets Indicates the status of CRC checking. If this field is 1, MAC receiver does not check the CRC field in the received packets. When this field becomes 0, the MAC receiver always checks the CRC field in the received packets. 0b - Enabled 1b - Disabled
15-14	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
13-0 GPSL	Giant Packet Size Limit Declares the status of a packet based on its size. If the received packet size (in bytes) is greater than the value programmed in this field, MAC declares the received packet as a giant packet. The value programmed in this field must be greater than or equal to 1,518 bytes. Also, any other programmed value is considered as 1,518 bytes. For VLAN-tagged packets, MAC adds 4 bytes to the programmed value. If MAC_VLAN_Tag_Ctrl[EDVLP] = 1, MAC adds 8 bytes to the programmed value for double VLAN-tagged packets. The value of GPSL is valid if MAC_Configuration[GPSLCE] = 1.

72.18.4 MAC Packet Filter (MAC_Packet_Filter)

Offset

Register	Offset
MAC_Packet_Filter	8h

Function

Contains the filter controls for receiving packets.

Some of the controls from this register go to MAC's address check block that performs the first level of address filtering. The second level of filtering is performed on the incoming packet based on other controls such as pass bad packets and pass control packets.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				HPF	SAF	SAIF	PCF			DBF	PM	DAIF	HMC	HUC	PR
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 RA	Receive All Indicates whether the received packets are enabled or disabled. • If this field is 1, the MAC Receiver module passes all the received packets to the application, irrespective of their passing the address filter. The result (pass or fail) of the SA or DA filtering is updated in the corresponding field of the receive status word. • If this field is 0, the module passes only those packets to the application that pass the SA or DA address filter. 0b - Receive All is disabled 1b - Receive All is enabled
30-22 —	Reserved
21 DNTU	Drop Non-TCP/UDP Over IP Packets Indicates whether MAC drops or forwards non-TCP/UDP protocols over IP packets. • If this field is 1, MAC drops these protocols over IP packets. MAC forwards only those packets that the layer 4 filter processes. • If this field is 0, MAC forwards all these protocols over IP packets. 0b - Forwards 1b - Drops
20 IPFE	Layer 3 and Layer 4 Filter Enable Indicates the status of layer 3 and layer 4 filters. • If this field is 1, MAC drops packets that do not match the enabled layer 3 and layer 4 filters. If these two filters are not enabled for matching, this field does not have any effect. • If this field is 0, MAC forwards all the packets irrespective of the match status of the layer 3 and layer 4 fields. 0b - Disabled 1b - Enabled
19-17 —	Reserved
16 VTFE	VLAN Tag Filter Enable Indicates the status of VLAN tag filter. • If this field is 1, MAC drops the VLAN-tagged packets that do not match the VLAN tag. • If this field is 0, MAC forwards all the packets irrespective of the match status of the VLAN tag. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-11 —	Reserved
10 HPF	Hash Or Perfect Filter Indicates the status of hash or perfect filter. • If this field is 1, the address filter passes a packet if it matches either perfect filtering or hash filtering as defined by the HMC or HUC fields of this register. • If this field is 0 and HUC = 1 or HMC = 1, the packet passes only if it matches the hash filter. 0b - Disabled 1b - Enabled
9 SAF	Source Address Filter Enable Indicates the status of SA filtering. • If this field is 1, MAC compares the SA field of the received packets with the values programmed in the enabled SA registers. If the comparison fails, MAC drops the packet. • When this field becomes 0, MAC forwards the received packet to the application with an updated field value, depending on the SA address comparison. **NOTE** According to the IEEE specification, bit 47 of the SA field is reserved. However, MAC compares all the 48 bits. You must consider this when programming the MAC address registers for SA. 0b - Disabled 1b - Enabled
8 SAIF	SA Inverse Filtering Indicates the status of SA inverse filtering. • If this field is 1, the address check block operates in the inverse filtering mode for SA address comparison. If the SA of a packet matches the values programmed in the SA registers, it is marked as failing the SA address filter. • If this field becomes 0 and the SA of a packet does not match the values programmed in the SA registers, the SA is marked as failing the SA address filter. 0b - Disabled 1b - Enabled
7-6 PCF	Pass Control Packets Controls the forwarding of all the control packets (including unicast and multicast pause packets). 00b - MAC filters all the control packets from reaching the application 01b - MAC forwards all the control packets, except pause packets, to the application even if they fail the address filter

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - MAC forwards all the control packets to the application even if they fail the address filter 11b - MAC forwards all the control packets that pass the address filter
5 DBF	Disable Broadcast Packets Enables or disables broadcast packets. • If this field is 1, AFM blocks all the incoming broadcast packets. In addition, it overrides all the other filter settings. • If this field is 0, AFM passes all the received broadcast packets. 0b - Enabled 1b - Disabled
4 PM	Pass All Multicast Enables or disables the passing of received packets. • If this field is 1, it indicates that all the received packets with a multicast destination address (first bit in the destination address field is 1) are passed. • If this field is 0, filtering of multicast packets depends on the settings of the HMC field. 0b - Disabled 1b - Enabled
3 DAIF	DA Inverse Filtering Indicates the status of DA inverse filtering. • If this field is 1, the Address Check block operates in Inverse Filtering mode for the DA address comparison of both unicast and multicast packets. • If this field is 0, normal filtering of packets is performed. 0b - Disabled 1b - Enabled
2 HMC	Hash Multicast Indicates the status of hast multicast. • If this field is 1, MAC performs the destination address filtering of received multicast packets according to the hash table. • If this field is 0, MAC performs the perfect destination address filtering for multicast packets, that is, it compares the DA field with the values programmed in DA registers. 0b - Disabled 1b - Enabled
1 HUC	Hash Unicast Indicates the status of hash unicast.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, MAC performs the destination address filtering of unicast packets according to the hash table. • When this field becomes 0, MAC performs a perfect destination address filtering for unicast packets, that is, it compares the DA field with the values you program in the MAC Address registers. 0b - Disabled 1b - Enabled
0 PR	Promiscuous Mode Indicates the status of Promiscuous mode. If this field is 1, the address filtering module passes all the incoming packets irrespective of the destination or source address. MAC clears the SA or DA filter fail status fields of the receive status word when PR = 1. 0b - Disabled 1b - Enabled

72.18.5 MAC Watchdog Timeout (MAC_Watchdog_Timeout)

Offset

Register	Offset
MAC_Watchdog_Timeout	Ch

Function

Controls the watchdog timeout for received packets.

Diagram

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

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

Fields

Field	Function
31-9 —	Reserved
8 PWE	Programmable Watchdog Enable Indicates the status of programmable watchdog. • If this field is 1 and MAC_Configuration[WD] is 0, the WTO field of this register is used as watchdog timeout for a received packet. • If this field is 0, MAC_Configuration[WD] and MAC_Configuration[JE] control the watchdog timeout for a received packet. 0b - Disabled 1b - Enabled
7-4 —	Reserved
3-0 WTO	Watchdog Timeout Indicates the size of watchdog timer. If PWE = 1 and MAC_Configuration[WD] = 0, this field is used as watchdog timeout for a received packet. If the length of a received packet exceeds the value of this field, the packet terminates as an error packet. **NOTE** If PWE = 1, the value in this field must exceed 1,522 (05F2h). Otherwise, the IEEE 802.3-specified valid tagged packets are declared as error packets and dropped. 0000b - 2 KB 0001b - 3 KB 0010b - 4 KB 0011b - 5 KB 0100b - 6 KB 0101b - 7 KB 0110b - 8 KB 0111b - 9 KB 1000b - 10 KB 1001b - 11 KB 1010b - 12 KB 1011b - 13 KB 1100b - 14 KB 1101b - 15 KB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1110b - 16383 bytes 1111b - Reserved

72.18.6 MAC Hash Table First 32 Bits (MAC_Hash_Table_Reg0)

Offset

Register	Offset
MAC_Hash_Table_Reg0	10h

Function

Contains the first 32 bits of the hash table, when the total width of the hash table is 128 bits or 256 bits.

The hash table is used for group address filtering, for which the content of the destination address in the incoming packet is passed through the CRC logic and the bits of Receive CRC Error Packets (Rx_CRC_Error_Packets) are used to index the contents of the hash table as follows:

- 6 bits when you have a 64-bit hash
- 7 bits when you have a 128-bit hash
- 8 bits when you have a 256-bit hash

The most-significant bits determine the Hash Table register to be used, and the least-significant 5 bits determine the bit within the register. For example, a hash value of 10_0000b (in 64-bit hash) selects bit 0 of MAC Hash Table Second 32 Bits (MAC_Hash_Table_Reg1).

Perform these steps to calculate the hash value of the destination address:

1. Calculate the 32-bit CRC for DA (see section 3.2.8 in IEEE 802.3 for the steps to calculate CRC32).
2. Perform bit-wise reversal for the value obtained in step 1.
3. Use the upper 6, 7, or 8 bits from the value obtained in step 2.

If this field is 1, the packet is accepted. Otherwise, it is rejected. If MAC_Packet_Filter[PM] = 1, all multicast packets are accepted regardless of the multicast hash values.

If this register is configured to be double-synchronized with the (G)MII clock domain, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the Hash Table registers are written to.

If double-synchronization is enabled, consecutive writes to this register must be performed after at least four clock cycles in the destination clock domain.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HT31T0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HT31T0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 HT31T0	MAC Hash Table First 32 Bits Contains the first 32 bits [31:0] of the hash table.

72.18.7 MAC Hash Table Second 32 Bits (MAC_Hash_Table_Reg1)

Offset

Register	Offset
MAC_Hash_Table_Reg1	14h

Function

Contains the second 32 bits of the hash table, when the width of the hash table is 128 or 256 bits.

The hash table is used for group address filtering, for which the content of the destination address in the incoming packet is passed through the CRC logic and the bits of Receive CRC Error Packets (Rx_CRC_Error_Packets) are used to index the contents of the hash table as follows:

- 6 bits when you have a 64-bit hash
- 7 bits when you have a 128-bit hash
- 8 bits when you have a 256-bit hash

The most-significant bits determine the Hash Table register to be used, and the least-significant 5 bits determine the bit within the register. For example, a hash value of 10_0000b (in 64-bit hash) selects bit 0 of MAC Hash Table Second 32 Bits (MAC_Hash_Table_Reg1).

Perform these steps to calculate the hash value of the destination address:

1. Calculate the 32-bit CRC for DA (see section 3.2.8 in IEEE 802.3 for the steps to calculate CRC32).
2. Perform bit-wise reversal for the value obtained in step 1.
3. Take the upper 6, 7, or 8 bits from the value obtained in step 2.

If the corresponding bit value of this register is 1'b1, the packet is accepted. Otherwise, it is rejected. If MAC_Packet_Filter[PM] = 1, all multicast packets are accepted regardless of the multicast hash values.

If this register is configured to be double-synchronized with the (G)MII clock domain, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the Hash Table registers are written to.

If double-synchronization is enabled, consecutive writes to this register must be performed after at least four clock cycles in the destination clock domain.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HT63T32															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HT63T32															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 HT63T32	MAC Hash Table Second 32 Bits Contains the second 32 bits [63:32] of the hash table.

72.18.8 MAC VLAN Tag (MAC_VLAN_Tag)

Offset

Register	Offset
MAC_VLAN_Tag	50h

Function

Identifies the IEEE 802.1Q VLAN type packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EIVLRXS	Reserved	EIVLS		ERIVLT	EDVLP	VTHM	EVLRXS	Reserved	EVLS		DOVLTC	ERSVLM	ESVL	VTIM	ETV
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31 EIVLRXS	Enable Inner VLAN Tag In Receive Status Indicates whether the inner VLAN tag in receive status is enabled or disabled. • If this field is 1, MAC provides the inner VLAN tag in the receive status. • If this field is 0, MAC does not provide the inner VLAN tag in the receive status. 0b - Disabled 1b - Enabled
30 —	Reserved
29-28 EIVLS	Enable Inner VLAN Tag Stripping Indicates the stripping operation on the inner VLAN tag in a received packet. The field enables or disables inner VLAN tag stripping on receive. 00b - Do not strip 01b - Strip if VLAN filter passes 10b - Strip if VLAN filter fails 11b - Always strip
27 ERIVLT	Enable Inner VLAN Tag Comparison Enables or disables the inner VLAN tag. • If ERIVLT = VTHM = EDVLP = 1, the EMAC receiver enables the VLAN hash filtering operation on the inner VLAN tag (if present). • If ERIVLT = 0 and VTHM = 1, the EMAC receiver enables the VLAN hash filtering operation on the outer VLAN tag (if present). ERSVLM and DOVLTC determine which VLAN type is enabled for filtering. 0b - Disabled 1b - Enabled
26 EDVLP	Enable Double VLAN Processing Enables or disables double VLAN processing. • If this field is 1, MAC enables processing of up to two VLAN tags on transmit and receive (if present). • If this field is 0, MAC enables processing of up to one VLAN tag on transmit and receive (if present). 0b - Disabled 1b - Enabled
25 VTHM	VLAN Tag Hash Table Match Indicates the status of VLAN tag hash table match.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, the most-significant four bits of CRC-32 of VLAN tag are used to index the content of MAC VLAN Hash Table (MAC_VLAN_Hash_Table). See VLAN filtering for details. • If MAC_VLAN_Hash_Table[VLHT] = 1, corresponding to the index, it indicates that the Ethernet packet matches the VLAN hash table. See VLAN filtering for details. • If ETV = 1, the CRC of the 12-bit VLAN identifier (VID) is used for comparison. When the ETV field becomes 0, the CRC of the 16-bit VLAN tag is used for comparison. • If VTHM = 1, the VLAN hash match operation is not performed. 0b - Disabled 1b - Enabled
24 EVLRXS	Enable VLAN Tag In Receive Status Indicates whether the VLAN tag in receive status is enabled. • If this field is 1, MAC provides the outer VLAN tag in the receive status. • If this field is 0, MAC does not provide the outer VLAN tag in receive status. 0b - Disabled 1b - Enabled
23 —	Reserved
22-21 EVLS	Enable VLAN Tag Stripping Indicates the stripping operation on the outer VLAN tag in received packets. The field enables or disables VLAN tag stripping on receive. 00b - Do not strip 01b - Strip if VLAN filter passes 10b - Strip if VLAN filter fails 11b - Always strip
20 DOVLTC	Disable VLAN Type Check Enables or disables VLAN type check for VLAN hash filtering. • If this field is 1, the MAC VLAN hash filter matches or filters the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. • If this field is 0, the MAC VLAN hash filter does not check whether the VLAN tag that the ERIVLT field specifies is of type S-VLAN or C-VLAN. 0b - Enabled 1b - Disabled
19 ERSVLM	Enable Receive S-VLAN Match Enables or disables receive S-VLAN match for VLAN hash filtering.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, the MAC receiver enables VLAN hash filtering or S-VLAN matching (type = 88A8h) packets. • If this field is 0, the MAC receiver enables VLAN hash filtering or matching for C-VLAN (type = 8100h) packets. The ERIVLT field determines the VLAN tag position considered for VLAN hash filtering or matching. 0b - Disabled 1b - Enabled
18 ESVL	Enable S-VLAN Enables or disables S-VLAN. If this field is 1, the MAC transmitter and receiver consider the S-VLAN packets (type = 88A8h) as valid VLAN-tagged packets. 0b - Disabled 1b - Enabled
17 VTIM	VLAN Tag Inverse Match Enable Enables or disables VLAN tag inverse matching. • If this field is 1, it enables the VLAN tag inverse matching. The packets without matching VLAN tag are marked as matched. • If this field is 0, it enables the VLAN tag perfect matching. The packets with matched VLAN tag are marked as matched. 0b - Disabled 1b - Enabled
16 ETV	Enable Tag For VLAN Enables or disables 12-bit VLAN tag comparison for VLAN hash filtering. • If this field is 1, a 12-bit VLAN identifier is used for VLAN hash filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag in the received VLAN-tagged packet are used for hash-based VLAN filtering. • When this field becomes 0, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for VLAN hash filtering. 0b - Disabled 1b - Enabled
15-0 VL	VLAN Tag Identifier for Receive Packets Contains the 802.1Q VLAN tag to identify the VLAN packets. This VLAN tag identifier is compared to the 15th and 16th bytes of the packets being received for VLAN packets. The following list describes the bits of this field:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 15:13 — User priority • 12 — Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • 11:0 — VID field of VLAN tag When ETV = 1, only the VID is used for comparison. When VL = 0, and ETV = 1, MAC does not check the 15th and 16th bytes for VLAN tag comparison and declares all packets with Type field value of 8100h or 88a8h as VLAN packets.

72.18.9 MAC VLAN Tag Control (MAC_VLAN_Tag_Ctrl)

Offset

Register	Offset
MAC_VLAN_Tag_Ctrl	50h

Function

Is the redefined version of the MAC_VLAN_Tag register.

This register provides the control and addressing fields required for indirect access to the MAC_VLAN_Tag_Filter registers. It also contains the address offset, command type, and busy bit for CSR access of the MAC VLAN Hash Filter registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EIVLRXS	Reserved	EIVLS		ERIVLT	EDVLP	VTHM	EVLRXS	Reserved	EVLS		DOVLTC	ERSVLM	ESVL	VTIM	ETV
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31 EIVLRXS	Enable Inner VLAN Tag In Receive Status Indicates whether the inner VLAN tag in receive status is enabled or disabled. • If this field is 1, MAC provides the inner VLAN tag in the receive status.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 0, MAC does not provide the inner VLAN tag in the receive status. 0b - Disabled 1b - Enabled
30 —	Reserved
29-28 EIVLS	Enable Inner VLAN Tag Stripping Indicates the stripping operation on the inner VLAN tag in a received packet. The field enables or disables inner VLAN tag stripping on receive. 00b - Do not strip 01b - Strip if VLAN filter passes 10b - Strip if VLAN filter fails 11b - Always strip
27 ERIVLT	Enable Inner VLAN Tag Comparison Enables or disables the inner VLAN tag. • If ERIVLT = VTHM = EDVLP = 1, the EMAC receiver enables the VLAN hash filtering operation on the inner VLAN tag (if present). • If ERIVLT = 0 and VTHM = 1, the EMAC receiver enables the VLAN hash filtering operation on the outer VLAN tag (if present). ERSVLM and DOVLTC determine which VLAN type is enabled for filtering. 0b - Disabled 1b - Enabled
26 EDVLP	Enable Double VLAN Processing Enables or disables double VLAN processing. • If this field is 1, MAC enables processing of up to two VLAN tags on transmit and receive (if present). • If this field is 0, MAC enables processing of up to one VLAN tag on transmit and receive (if present). 0b - Disabled 1b - Enabled
25 VTHM	VLAN Tag Hash Table Match Indicates the status of VLAN tag hash table match. • If this field is 1, the most-significant four bits of CRC-32 of VLAN tag are used to index the content of MAC VLAN Hash Table (MAC_VLAN_Hash_Table). See VLAN filtering for details. • If MAC_VLAN_Hash_Table[VLHT] = 1, corresponding to the index, it indicates that the Ethernet packet matches the VLAN hash table. See VLAN filtering for details.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If ETV = 1, the CRC of the 12-bit VLAN identifier (VID) is used for comparison. When the ETV field becomes 0, the CRC of the 16-bit VLAN tag is used for comparison. • If VTHM = 1, the VLAN hash match operation is not performed. 0b - Disabled 1b - Enabled
24 EVLRXS	Enable VLAN Tag In Receive Status Indicates whether the VLAN tag in receive status is enabled. • If this field is 1, MAC provides the outer VLAN tag in the receive status. • If this field is 0, MAC does not provide the outer VLAN tag in receive status. 0b - Disabled 1b - Enabled
23 —	Reserved
22-21 EVLS	Enable VLAN Tag Stripping Indicates the stripping operation on the outer VLAN tag in received packets. The field enables or disables VLAN tag stripping on receive. 00b - Do not strip 01b - Strip if VLAN filter passes 10b - Strip if VLAN filter fails 11b - Always strip
20 DOVLTC	Disable VLAN Type Check Enables or disables VLAN type check for VLAN hash filtering. • If this field is 1, the MAC VLAN hash filter matches or filters the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. • If this field is 0, the MAC VLAN hash filter does not check whether the VLAN tag that the ERIVLT field specifies is of type S-VLAN or C-VLAN. 0b - Enabled 1b - Disabled
19 ERSVLM	Enable Receive S-VLAN Match Enables or disables receive S-VLAN match for VLAN hash filtering. • If this field is 1, the MAC receiver enables VLAN hash filtering or S-VLAN matching (type = 88A8h) packets. • If this field is 0, the MAC receiver enables VLAN hash filtering or matching for C-VLAN (type = 8100h) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The ERIVLT field determines the VLAN tag position considered for VLAN hash filtering or matching. 0b - Disabled 1b - Enabled
18 ESVL	Enable S-VLAN Enables or disables S-VLAN. If this field is 1, the MAC transmitter and receiver consider the S-VLAN packets (type = 88A8h) as valid VLAN-tagged packets. 0b - Disabled 1b - Enabled
17 VTIM	VLAN Tag Inverse Match Enable Enables or disables VLAN tag inverse matching. • If this field is 1, it enables the VLAN tag inverse matching. The packets without matching VLAN tag are marked as matched. • If this field is 0, it enables the VLAN tag perfect matching. The packets with matched VLAN tag are marked as matched. 0b - Disabled 1b - Enabled
16 ETV	Enable Tag For VLAN Enables or disables 12-bit VLAN tag comparison for VLAN hash filtering. • If this field is 1, a 12-bit VLAN identifier is used for VLAN hash filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag in the received VLAN-tagged packet are used for hash-based VLAN filtering. • When this field becomes 0, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for VLAN hash filtering. 0b - Disabled 1b - Enabled
15-4 —	Reserved
3-2 OFS	Offset Holds the address offset of the MAC VLAN Tag Filter register that the application tries to access. The width of this field depends on the number of enabled MAC VLAN Tag registers.
1 CT	Command Type Specifies whether the current register access indicates a read or a write operation. It indicates a read operation if this field is 1 and a write operation when the field is 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Write operation 1b - Read operation
0 OB	Operation Busy Indicates the operation busy status. • This field becomes 1 to initiate a read or write command for an indirect access to the Per VLAN Tag Filter register. • The field becomes 0 when the read or write command to Per VLAN Tag Filter indirect access register completes. The next indirect register access can be initiated only after the field resets. During a write operation, the field becomes 0 only after the data is written into a MAC_VLAN_Tag_Filter register. During a read operation, the data must be read from MAC VLAN Tag Data (MAC_VLAN_Tag_Data) only after this field becomes 0. 0b - Disabled 1b - Enabled

72.18.10 MAC VLAN Tag Data (MAC_VLAN_Tag_Data)

Offset

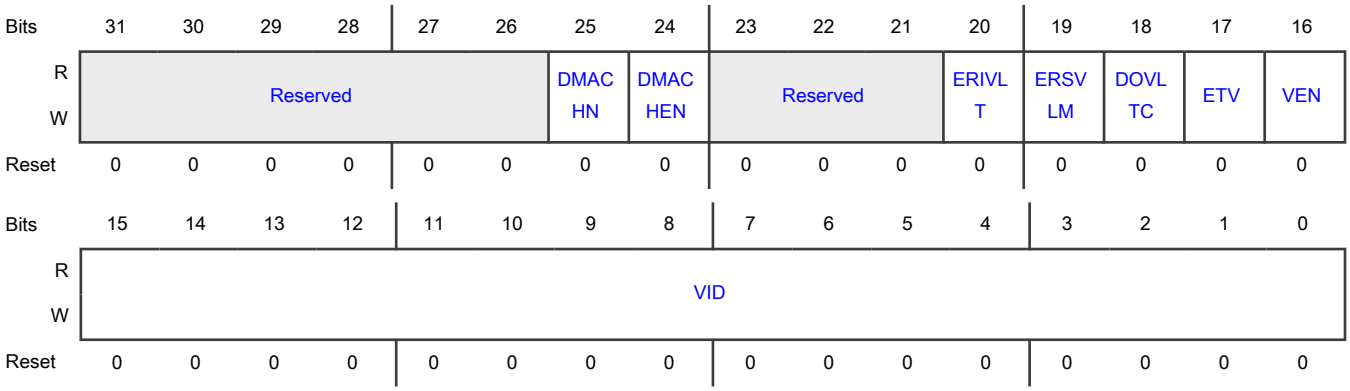
Register	Offset
MAC_VLAN_Tag_Data	54h

Function

Holds the read and write data for indirect access to the Per VLAN Tag registers.

- During read access, this register contains valid read data only after MAC_VLAN_Tag_Ctrl[OB] becomes 0.
- During write access, this register must become valid before you write 1 to MAC_VLAN_Tag_Ctrl[OB].

Diagram



Fields

Field	Function
31-26 —	Reserved
25 DMACHN	DMA Channel Number Indicates the DMA channel number to which the VLAN-tagged frame is to be routed if it passes the VLAN tag filter programmed using this field. If the routing based on VLAN tag filter is not necessary, this field does not need to be programmed.
24 DMACHEN	DMA Channel Number Enable Enables or disables the DMA channel number value programmed using the DMACHN field. If this field is 0, the routing does not occur based on the VLAN filter result. The frame is routed on the basis of DA-based DMA channel routing. 0b - Disabled 1b - Enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Enables or disables inner VLAN tag comparison. • This field is valid only when the double VLAN tag enable of the filter is set. • If this field and the EDVLP field is 1, the MAC receiver enables operation on the inner VLAN tag (if present). • If this field is 0, the MAC receiver enables operation on the outer VLAN tag (if present). 0b - Disabled 1b - Enabled
19 ERSVLM	Enable S-VLAN Match Enables or disables S-VLAN match for received frames. This field is valid only when the VLAN tag enable of the filter is set. • If this field is 1, the MAC receiver enables filtering or matching for S-VLAN (type = 88A8h) packets. • If this field is 0, the MAC receiver enables filtering or matching for C-VLAN (type = 8100h) packets. 0b - Disabled 1b - Enabled
18 DOVLTC	Disable VLAN Type Comparison Enables or disables VLAN type comparison. This field is valid only when the VEN field is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, MAC does not check whether the VLAN tag that ERIVLT specifies is of type S-VLAN or C-VLAN. • If this field is 0, MAC filters or matches the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. 0b - Enabled 1b - Disabled
17 ETV	VLAN Comparison Indicates 12-bit or 16-bit VLAN comparison. • This field is valid only when the VEN field is 1. • When the value of this field is 1, a 12-bit VLAN identifier is used for filter and comparison instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16-bit VLAN comparison 1b - 12-bit VLAN comparison
16 VEN	VLAN Tag Enable Enables or disables the VLAN tag. • If this field is 1, MAC compares the VLAN tag of the received packet with the VLAN tag ID. • If this field is 0, no comparison is performed irrespective of the programming of the other fields. 0b - Disabled 1b - Enabled
15-0 VID	VLAN Tag ID Holds the VLAN tag value that MAC uses for perfect comparison. The field is valid when VEN = 1.

72.18.11 MAC VLAN Tag Filter 0 (MAC_VLAN_Tag_Filter0)

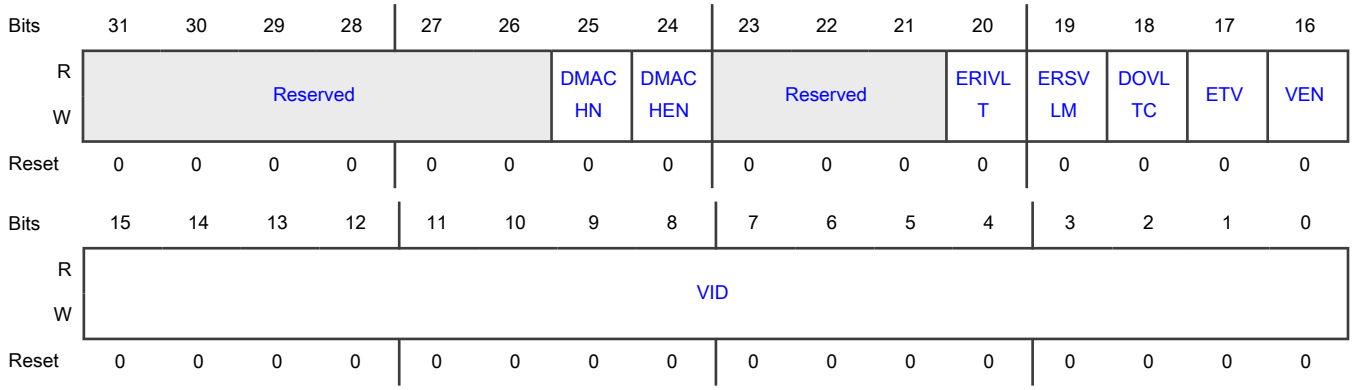
Offset

Register	Offset
MAC_VLAN_Tag_Filter0	54h

Function

Contains VLAN tag filter 0 control information.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 DMACHN	DMA Channel Number Indicates the DMA channel number to which the VLAN-tagged frame is to be routed if it passes the VLAN tag filter programmed using this field. If the routing based on VLAN tag filter is not necessary, this field does not need to be programmed.
24 DMACHEN	DMA Channel Number Enable Enables or disables the DMA channel number value programmed using the DMACHN field. If this field is 0, the routing does not occur based on the VLAN filter result. The frame is routed on the basis of DA-based DMA channel routing. 0b - Disabled 1b - Enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Enables or disables inner VLAN tag comparison. • This field is valid only when the double VLAN tag enable of the filter is set. • If this field and the EDVLP field is 1, the MAC receiver enables operation on the inner VLAN tag (if present). • If this field is 0, the MAC receiver enables operation on the outer VLAN tag (if present). 0b - Disabled 1b - Enabled
19	Enable S-VLAN Match

Table continues on the next page...

Table continued from the previous page...

Field	Function
ERSVLM	Enables or disables S-VLAN match for received frames. This field is valid only when the VLAN tag enable of the filter is set. • If this field is 1, the MAC receiver enables filtering or matching for S-VLAN (type = 88A8h) packets. • If this field is 0, the MAC receiver enables filtering or matching for C-VLAN (type = 8100h) packets. 0b - Disabled 1b - Enabled
18 DOVLTC	Disable VLAN Type Comparison Enables or disables VLAN type comparison. This field is valid only when the VEN field is 1. • If this field is 1, MAC does not check whether the VLAN tag that ERIVLT specifies is of type S-VLAN or C-VLAN. • If this field is 0, MAC filters or matches the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. 0b - Enabled 1b - Disabled
17 ETV	VLAN Comparison Indicates 12-bit or 16-bit VLAN comparison. • This field is valid only when the VEN field is 1. • When the value of this field is 1, a 12-bit VLAN identifier is used for filter and comparison instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16-bit VLAN comparison 1b - 12-bit VLAN comparison
16 VEN	VLAN Tag Enable Enables or disables the VLAN tag. • If this field is 1, MAC compares the VLAN tag of the received packet with the VLAN tag ID. • If this field is 0, no comparison is performed irrespective of the programming of the other fields. 0b - Disabled 1b - Enabled
15-0 VID	VLAN Tag ID Holds the VLAN tag value that MAC uses for perfect comparison. The field is valid when VEN = 1.

72.18.12 MAC VLAN Tag Filter 1 (MAC_VLAN_Tag_Filter1)

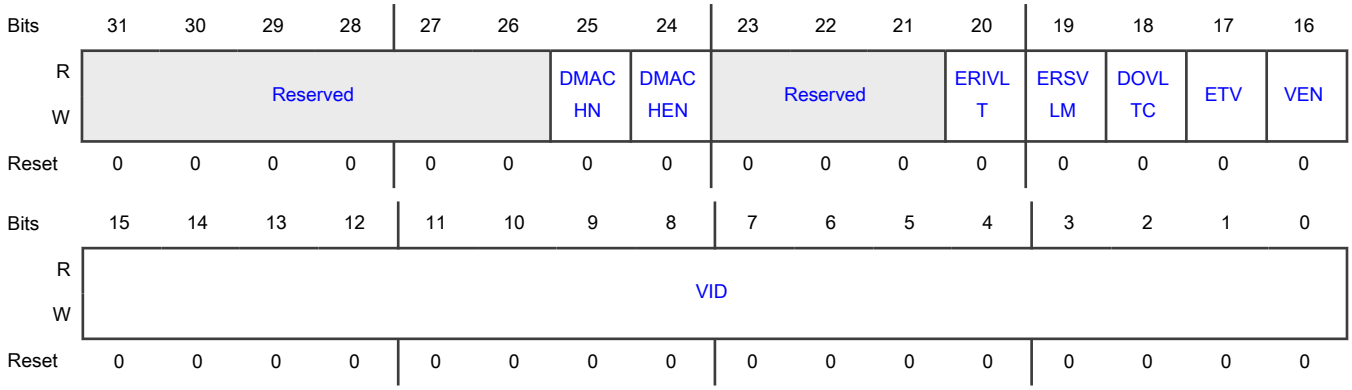
Offset

Register	Offset
MAC_VLAN_Tag_Filter1	54h

Function

Contains VLAN tag filter 1 control information.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 DMACHN	DMA Channel Number Indicates the DMA channel number to which the VLAN-tagged frame is to be routed if it passes the VLAN tag filter programmed using this field. If the routing based on VLAN tag filter is not necessary, this field does not need to be programmed.
24 DMACHEN	DMA Channel Number Enable Enables or disables the DMA channel number value programmed using the DMACHN field. If this field is 0, the routing does not occur based on the VLAN filter result. The frame is routed on the basis of DA-based DMA channel routing. 0b - Disabled 1b - Enabled
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 ERIVLT	Enable Inner VLAN Tag Enables or disables inner VLAN tag comparison. • This field is valid only when the double VLAN tag enable of the filter is set. • If this field and the EDVLP field is 1, the MAC receiver enables operation on the inner VLAN tag (if present). • If this field is 0, the MAC receiver enables operation on the outer VLAN tag (if present). 0b - Disabled 1b - Enabled
19 ERSVLM	Enable S-VLAN Match Enables or disables S-VLAN match for received frames. This field is valid only when the VLAN tag enable of the filter is set. • If this field is 1, the MAC receiver enables filtering or matching for S-VLAN (type = 88A8h) packets. • If this field is 0, the MAC receiver enables filtering or matching for C-VLAN (type = 8100h) packets. 0b - Disabled 1b - Enabled
18 DOVLTC	Disable VLAN Type Comparison Enables or disables VLAN type comparison. This field is valid only when the VEN field is 1. • If this field is 1, MAC does not check whether the VLAN tag that ERIVLT specifies is of type S-VLAN or C-VLAN. • If this field is 0, MAC filters or matches the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. 0b - Enabled 1b - Disabled
17 ETV	VLAN Comparison Indicates 12-bit or 16-bit VLAN comparison. • This field is valid only when the VEN field is 1. • When the value of this field is 1, a 12-bit VLAN identifier is used for filter and comparison instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16-bit VLAN comparison 1b - 12-bit VLAN comparison
16 VEN	VLAN Tag Enable Enables or disables the VLAN tag.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, MAC compares the VLAN tag of the received packet with the VLAN tag ID. • If this field is 0, no comparison is performed irrespective of the programming of the other fields. 0b - Disabled 1b - Enabled
15-0 VID	VLAN Tag ID Holds the VLAN tag value that MAC uses for perfect comparison. The field is valid when VEN = 1.

72.18.13 MAC VLAN Tag Filter 2 (MAC_VLAN_Tag_Filter2)

Offset

Register	Offset
MAC_VLAN_Tag_Filter2	54h

Function
Contains VLAN tag filter 2 control information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved						DMACHN	DMACHEN	Reserved			ERIVLT	ERSVLM	DOVLTC	ETV	VEN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Fields

Field	Function
31-26 —	Reserved
25 DMACHN	DMA Channel Number Indicates the DMA channel number to which the VLAN-tagged frame is to be routed if it passes the VLAN tag filter programmed using this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If the routing based on VLAN tag filter is not necessary, this field does not need to be programmed.
24 DMACHEN	DMA Channel Number Enable Enables or disables the DMA channel number value programmed using the DMACHN field. If this field is 0, the routing does not occur based on the VLAN filter result. The frame is routed on the basis of DA-based DMA channel routing. 0b - Disabled 1b - Enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Enables or disables inner VLAN tag comparison. • This field is valid only when the double VLAN tag enable of the filter is set. • If this field and the EDVLP field is 1, the MAC receiver enables operation on the inner VLAN tag (if present). • If this field is 0, the MAC receiver enables operation on the outer VLAN tag (if present). 0b - Disabled 1b - Enabled
19 ERSVLM	Enable S-VLAN Match Enables or disables S-VLAN match for received frames. This field is valid only when the VLAN tag enable of the filter is set. • If this field is 1, the MAC receiver enables filtering or matching for S-VLAN (type = 88A8h) packets. • If this field is 0, the MAC receiver enables filtering or matching for C-VLAN (type = 8100h) packets. 0b - Disabled 1b - Enabled
18 DOVLTC	Disable VLAN Type Comparison Enables or disables VLAN type comparison. This field is valid only when the VEN field is 1. • If this field is 1, MAC does not check whether the VLAN tag that ERIVLT specifies is of type S-VLAN or C-VLAN. • If this field is 0, MAC filters or matches the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. 0b - Enabled 1b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 ETV	VLAN Comparison Indicates 12-bit or 16-bit VLAN comparison. • This field is valid only when the VEN field is 1. • When the value of this field is 1, a 12-bit VLAN identifier is used for filter and comparison instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16-bit VLAN comparison 1b - 12-bit VLAN comparison
16 VEN	VLAN Tag Enable Enables or disables the VLAN tag. • If this field is 1, MAC compares the VLAN tag of the received packet with the VLAN tag ID. • If this field is 0, no comparison is performed irrespective of the programming of the other fields. 0b - Disabled 1b - Enabled
15-0 VID	VLAN Tag ID Holds the VLAN tag value that MAC uses for perfect comparison. The field is valid when VEN = 1.

72.18.14 MAC VLAN Tag Filter 3 (MAC_VLAN_Tag_Filter3)

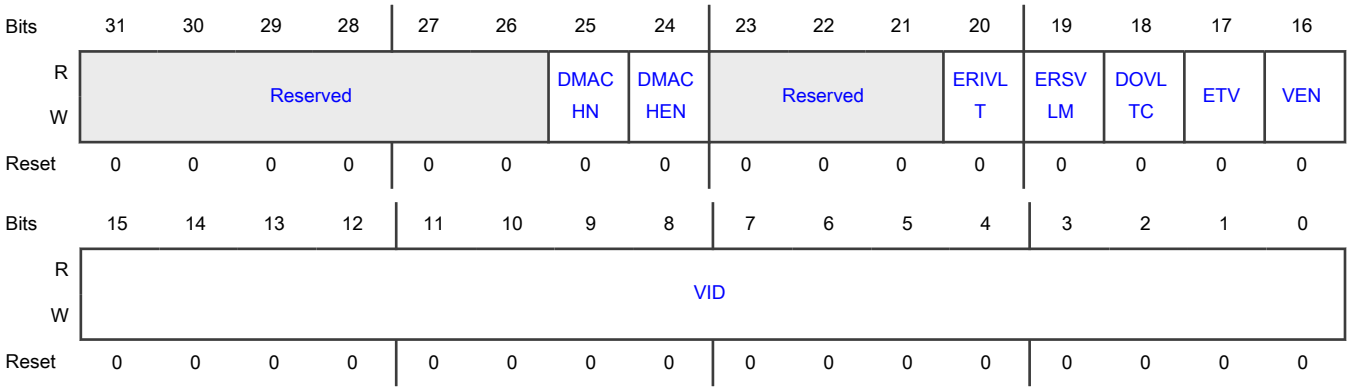
Offset

Register	Offset
MAC_VLAN_Tag_Filter3	54h

Function

Contains VLAN tag filter 3 control information.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 DMACHN	DMA Channel Number Indicates the DMA channel number to which the VLAN-tagged frame is to be routed if it passes the VLAN tag filter programmed using this field. If the routing based on VLAN tag filter is not necessary, this field does not need to be programmed.
24 DMACHEN	DMA Channel Number Enable Enables or disables the DMA channel number value programmed using the DMACHN field. If this field is 0, the routing does not occur based on the VLAN filter result. The frame is routed on the basis of DA-based DMA channel routing. 0b - Disabled 1b - Enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Enables or disables inner VLAN tag comparison. • This field is valid only when the double VLAN tag enable of the filter is set. • If this field and the EDVLP field is 1, the MAC receiver enables operation on the inner VLAN tag (if present). • If this field is 0, the MAC receiver enables operation on the outer VLAN tag (if present). 0b - Disabled 1b - Enabled
19 ERSVLM	Enable S-VLAN Match Enables or disables S-VLAN match for received frames. This field is valid only when the VLAN tag enable of the filter is set. • If this field is 1, the MAC receiver enables filtering or matching for S-VLAN (type = 88A8h) packets. • If this field is 0, the MAC receiver enables filtering or matching for C-VLAN (type = 8100h) packets. 0b - Disabled 1b - Enabled
18 DOVLTC	Disable VLAN Type Comparison Enables or disables VLAN type comparison. This field is valid only when the VEN field is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, MAC does not check whether the VLAN tag that ERIVLT specifies is of type S-VLAN or C-VLAN. • If this field is 0, MAC filters or matches the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. 0b - Enabled 1b - Disabled
17 ETV	VLAN Comparison Indicates 12-bit or 16-bit VLAN comparison. • This field is valid only when the VEN field is 1. • When the value of this field is 1, a 12-bit VLAN identifier is used for filter and comparison instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16-bit VLAN comparison 1b - 12-bit VLAN comparison
16 VEN	VLAN Tag Enable Enables or disables the VLAN tag. • If this field is 1, MAC compares the VLAN tag of the received packet with the VLAN tag ID. • If this field is 0, no comparison is performed irrespective of the programming of the other fields. 0b - Disabled 1b - Enabled
15-0 VID	VLAN Tag ID Holds the VLAN tag value that MAC uses for perfect comparison. The field is valid when VEN = 1.

72.18.15 MAC VLAN Hash Table (MAC_VLAN_Hash_Table)

Offset

Register	Offset
MAC_VLAN_Hash_Table	58h

Function

If MAC_VLAN_Tag_Ctrl[VTHM] = 1, the 16-bit VLAN hash table register is used for group address filtering based on the VLAN tag. For hash filtering, the content of the 16-bit VLAN tag or 12-bit VLAN ID (based on MAC_VLAN_Tag_Ctrl[ETV]) in the incoming packet is passed through the CRC logic. The upper four bits of the calculated hash value are used to index the contents of the VLAN hash table. For example, hash value of 1000b selects bit 8 of the VLAN hash table.

Perform these steps to calculate the hash value of the destination address:

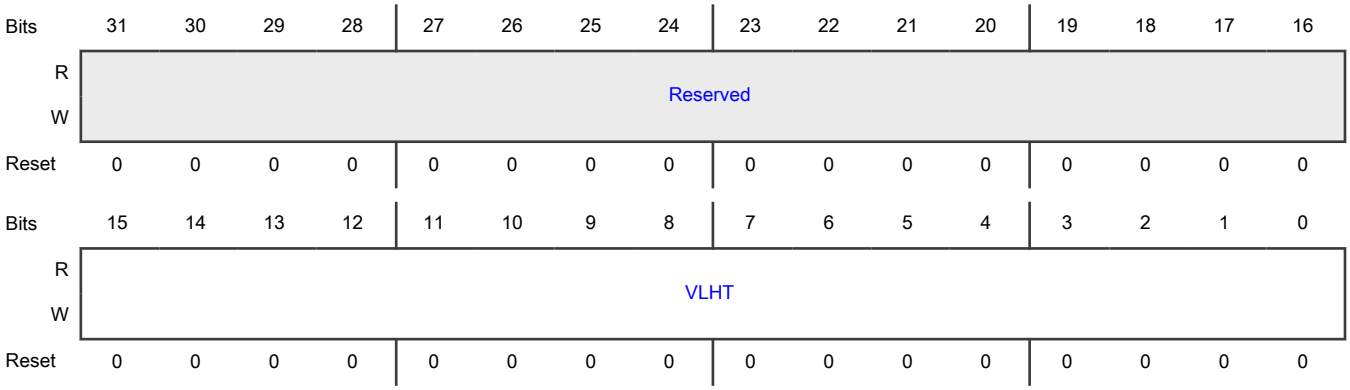
1. Calculate the 32-bit CRC for DA (see section 3.2.8 of IEEE 802.3 for steps to calculate CRC32).

2. Perform bit-wise reversal for the value obtained in step 1.
3. Take the upper 4 bits from the value obtained in step 2.

If this register is configured to be double-synchronized with the (G)MII clock domain, the synchronization is triggered only when bits [15:8] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the register are written to.

If double-synchronization is enabled, consecutive writes to this register must be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 VLHT	VLAN Hash Table Contains the 16-bit VLAN hash table.

72.18.16 MAC VLAN Inclusion Or Replacement (MAC_VLAN_Incl)

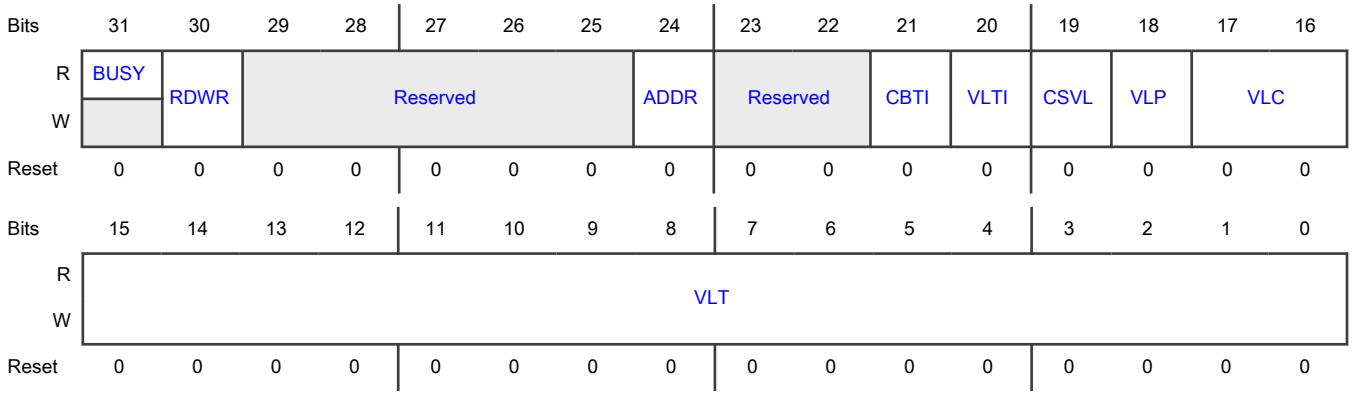
Offset

Register	Offset
MAC_VLAN_Incl	60h

Function

Contains the VLAN tag for insertion or replacement in the transmit packets and the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31 BUSY	Busy Indicates the status of the read or write operation of indirect access to the queue- or channel-specific VLAN inclusion register. The write operation to a register completes when this field becomes 0. For a read operation, the read data is valid when the field becomes 0. The application must make sure that this field becomes 0 before attempting subsequent accesses to this register. 0b - Busy status not detected 1b - Busy status detected
30 RDWR	Read Write Control Controls the read or write operation for indirectly accessing the queue- or channel-specific VLAN inclusion register. • If this field is 1, it indicates the write operation. • If this field is 0, it indicates the read operation. The value of this field is not impacted when the CBTI field becomes 0. 0b - Read operation of indirect access 1b - Write operation of indirect access
29-25 —	Reserved
24 ADDR	Address Selects one of the queue- or channel-specific VLAN inclusion registers for read or write access. The value of this field is not impacted when the CBTI field is 0.
23-22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
21 CBTI	Channel-Based Tag Insertion Indicates the status of channel-based tag insertion. If this field is 1: • The outer VLAN tag is inserted for all the packets that MAC transmits. • The tag value is taken from the queue- or channel-specific VLAN tag register. Also, the VLTI, VLP, VLC, and VLT fields of the register are ignored. • A write operation to byte 3 of this register initiates the read or write access to the indirect register. If this field is 0, the outer VLAN operation is based on the settings of the VLTI, VLP, VLC, and VLT fields of this register. 0b - Disabled 1b - Enabled
20 VLTI	VLAN Tag Input Indicates the status of the VLAN tag input. The value of this field being 1 indicates that the VLAN tag to be inserted or replaced in the transmit packet must be taken from the transmit descriptor. 0b - Disabled 1b - Enabled
19 CSVL	C-VLAN Or S-VLAN Indicates whether C-VLAN or S-VLAN is inserted or replaced in the transmitted packets. • If this field is 1, S-VLAN type (88A8h) is inserted or replaced in the 13th and 14th bytes of the transmitted packets. • If this field is 0, C-VLAN type (8100h) is inserted or replaced in the 13th and 14th bytes of the transmitted packets. 0b - C-VLAN 1b - S-VLAN
18 VLP	VLAN Priority Control Indicates the status of VLAN priority control. • If this field is 1, the control bits [17:16] are used for VLAN deletion, insertion, or replacement. • If this field is 0, the mti_vlan_ctrl_i control input is used and bits [17:16] are ignored. 0b - Disabled 1b - Enabled
17-16	VLAN Tag Control Contains values for VLAN tag control in transmit packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
VLC	MAC: • Removes the VLAN type (bytes 13 and 14) and VLAN tag (bytes 15 and 16) of all the transmitted packets with VLAN tags. • Replaces VLT in bytes 15 and 16 of all the VLAN-type transmitted packets (bytes 13 and 14 are 8100h or 88a8h). MAC inserts the following into the packet in the order shown: 1. Type value (8100h or 88A8h) into bytes 13 and 14 2. MAC_VLAN_Tag_Data[VLT] into bytes 15 and 16 This operation is performed on all the transmitted packets, irrespective of whether they already have a VLAN tag. **NOTE** Changes to this field take effect only on the start of a packet. If you write to this field when a packet is being transmitted, only the subsequent packets can use the updated value. That is, the current packet does not use the updated value. 00b - No VLAN tag deletion, insertion, or replacement 01b - VLAN tag deletion 10b - VLAN tag insertion 11b - VLAN tag replacement
15-0 VLT	VLAN Tag For Transmit Packets Contains the value of the VLAN tag to be inserted or replaced. This value must only be changed during the initialization phase or when the transmit lines are inactive. The following list describes the bits of this field: • Bits [15:13]: User priority • Bit 12: Canonical format indicator (CFI) or drop eligible indicator (DEI) • Bits [11:0]: MAC_VLAN_Tag_Data[VID]

72.18.17 MAC VLAN Inclusion 0 (MAC_VLAN_Incl0)

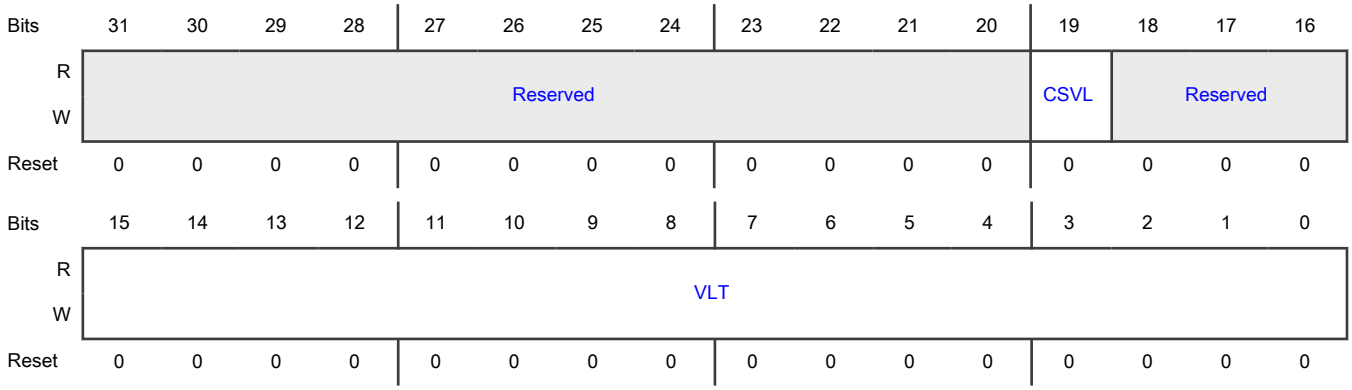
Offset

Register	Offset
MAC_VLAN_Incl0	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 0. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

72.18.18 MAC VLAN Inclusion 1 (MAC_VLAN_Incl1)

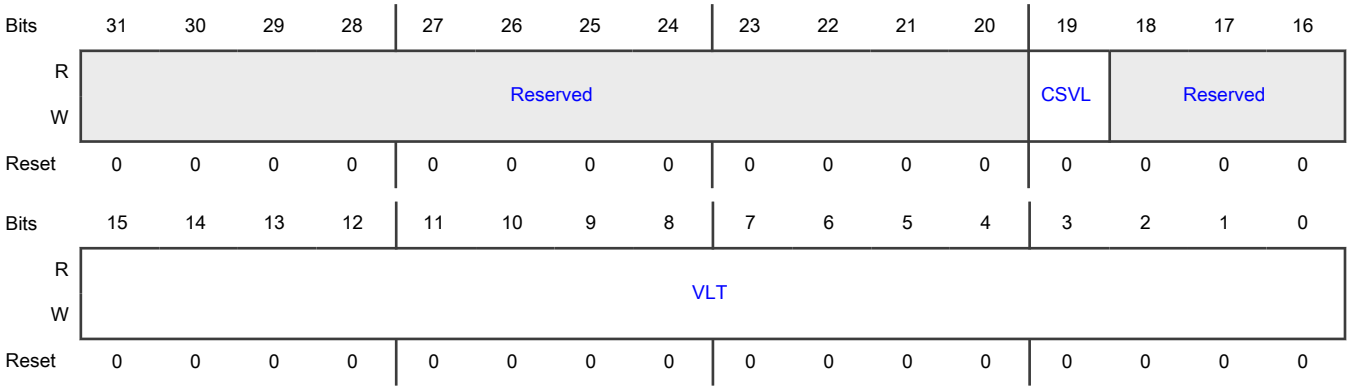
Offset

Register	Offset
MAC_VLAN_Incl1	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 1. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

72.18.19 MAC VLAN Inclusion 2 (MAC_VLAN_Incl2)

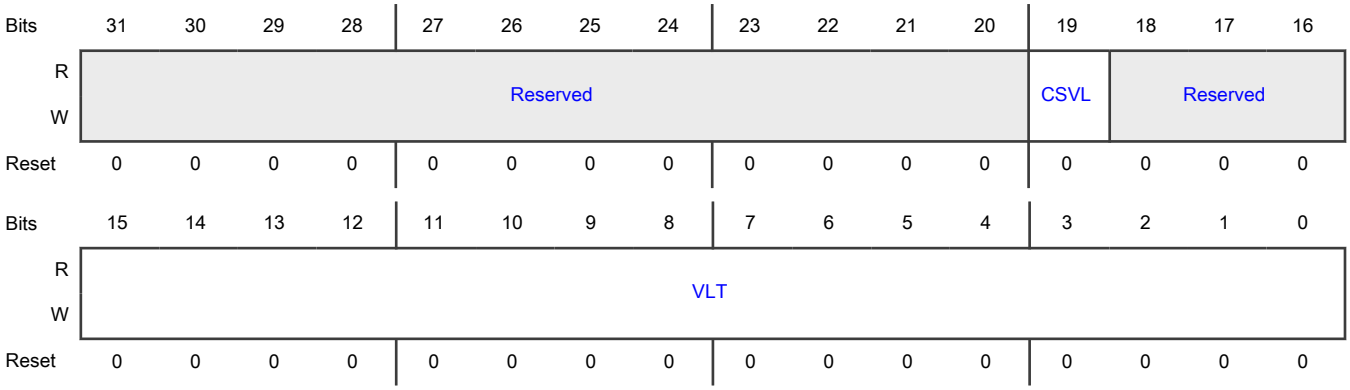
Offset

Register	Offset
MAC_VLAN_Incl2	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 2. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

72.18.20 MAC VLAN Inclusion 3 (MAC_VLAN_Incl3)

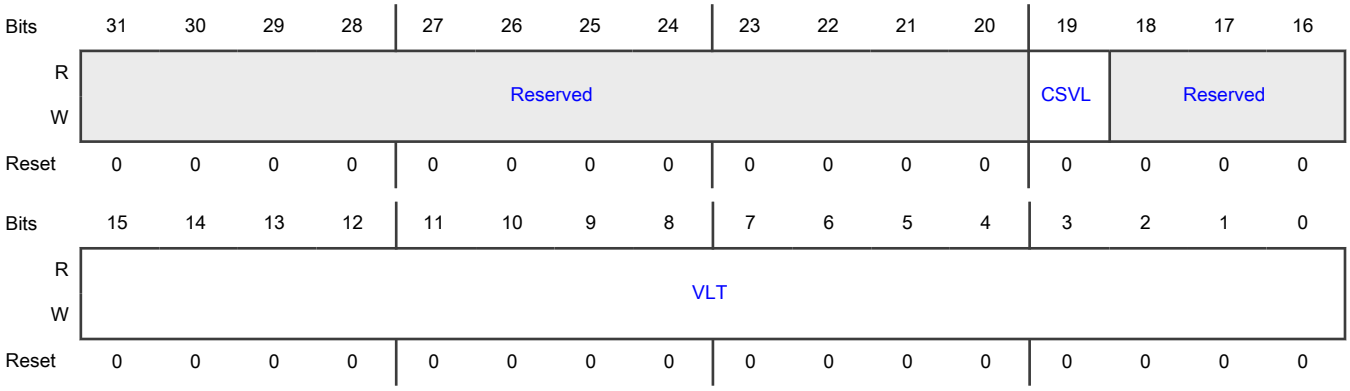
Offset

Register	Offset
MAC_VLAN_Incl3	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 3. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

72.18.21 MAC VLAN Inclusion 4 (MAC_VLAN_Incl4)

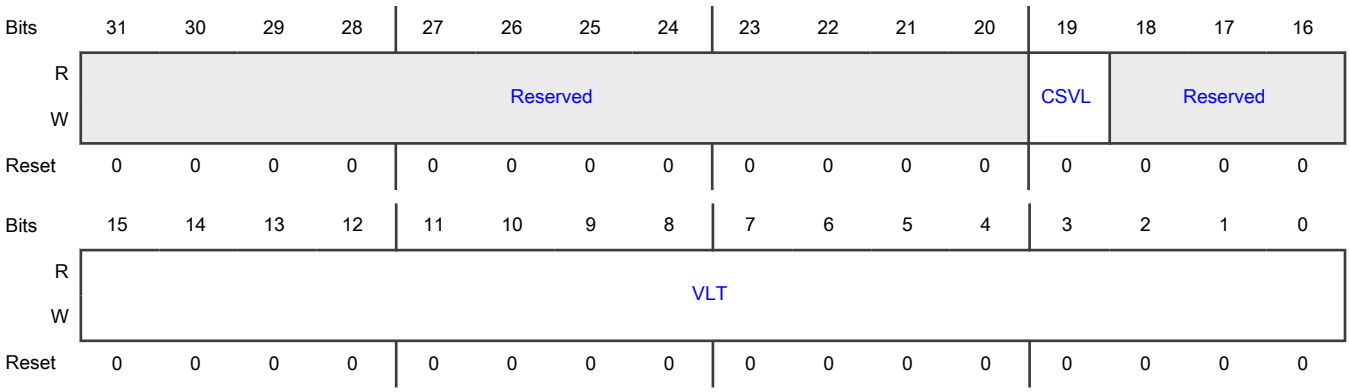
Offset

Register	Offset
MAC_VLAN_Incl4	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 4. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

72.18.22 MAC VLAN Inclusion 5 (MAC_VLAN_Incl5)

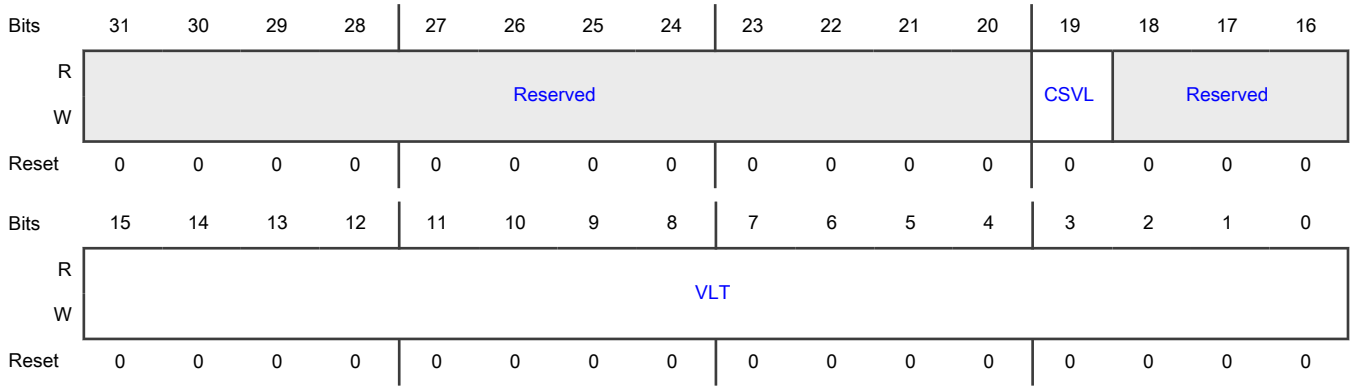
Offset

Register	Offset
MAC_VLAN_Incl5	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 5. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

72.18.23 MAC VLAN Inclusion 6 (MAC_VLAN_Incl6)

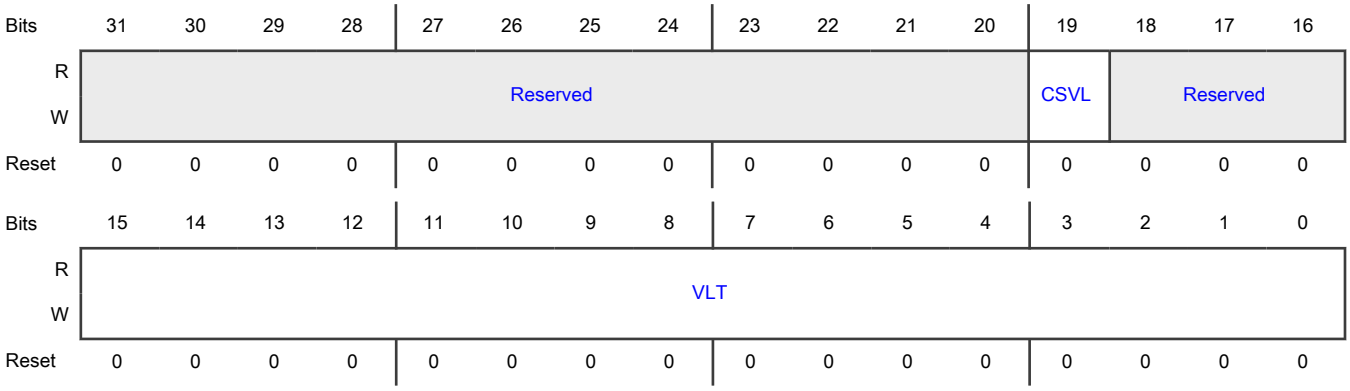
Offset

Register	Offset
MAC_VLAN_Incl6	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 6. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

72.18.24 MAC VLAN Inclusion 7 (MAC_VLAN_Incl7)

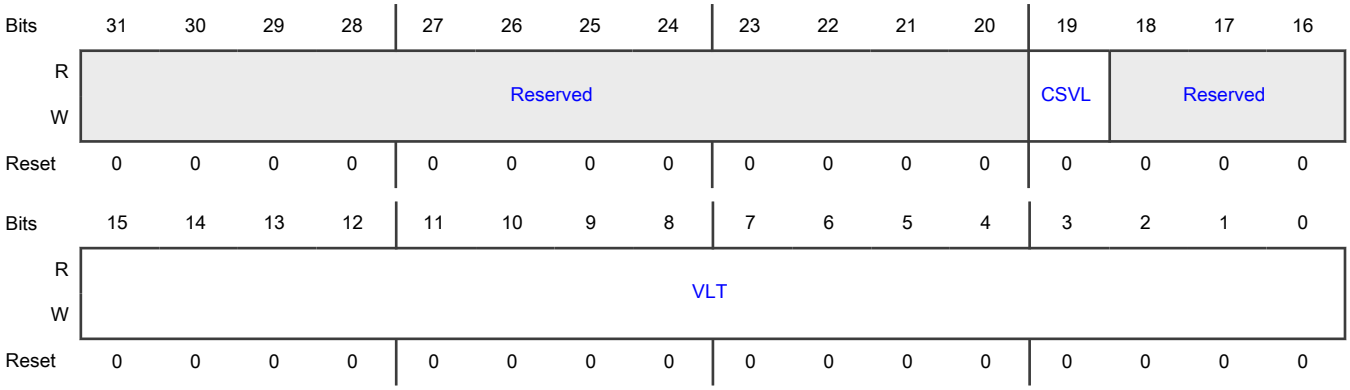
Offset

Register	Offset
MAC_VLAN_Incl7	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 7. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

72.18.25 Inner VLAN Tag Inclusion Or Replacement (MAC_Inner_VLAN_Incl)

Offset

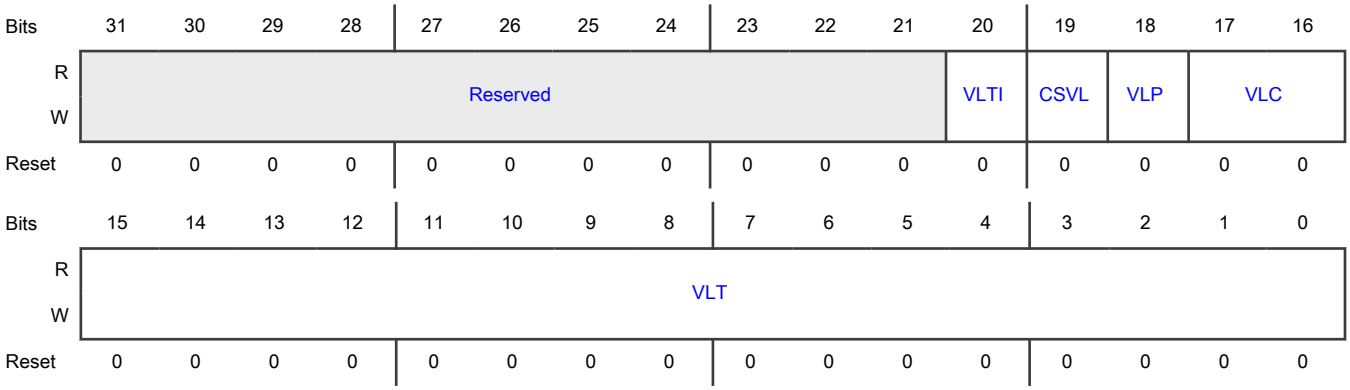
Register	Offset
MAC_Inner_VLAN_Incl	64h

Function

Contains:

- The inner VLAN tag to be inserted or replaced in the transmit packet
- The inner VLAN tag insertion controls

Diagram



Fields

Field	Function
31-21 —	Reserved
20 VLTI	VLAN Tag Input Indicates the status of VLAN tag input. If this field is 1, it indicates that the VLAN tag to be inserted or replaced in the transmit packet must be taken from the transmit descriptor.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
19 CSVL	C-VLAN Or S-VLAN Controls the insertion or replacement type in the 17th and 18th bytes of the transmitted packets. 0b - C-VLAN type (8100h) 1b - S-VLAN type (88A8h)
18 VLP	VLAN Priority Control Indicates the status of VLAN priority control. • If this field is 1, it's used for VLAN deletion, insertion, or replacement. • If this field is 0, the mti_vlan_ctrl_i control input is used and MAC_VLAN_Incl[VLC] is ignored. 0b - Disabled 1b - Enabled
17-16 VLC	VLAN Tag Control in Transmit Packets Indicates the value of VLAN tag control in transmit packets. The following list specifies these values: • 2'b00: No VLAN tag deletion, insertion, or replacement • 2'b01: VLAN tag deletion • 2'b10: VLAN tag insertion • 2'b11: VLAN tag replacement MAC: • Removes the VLAN type (bytes 17 and 18) and VLAN tag (bytes 19 and 20) of all the transmitted packets with VLAN tags • Inserts VLT in bytes 19 and 20 of the packet after inserting the Type value (8100h or 88a8h) in bytes 17 and 18. This operation is performed on all the transmitted packets, irrespective of whether they already have a VLAN tag. • Replaces VLT in bytes 19 and 20 for all the VLAN-type transmitted packets (bytes 17 and 18 are 8100h or 88a8h). **NOTE** Changes to this field take effect only on the start of a packet. If you write to the field when a packet is being transmitted, only the subsequent packets can use the updated value. That is, the current packet does not use the updated value. 00b - No VLAN tag deletion, insertion, or replacement 01b - VLAN tag deletion 10b - VLAN tag insertion

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - VLAN tag replacement
15-0 VLT	VLAN Tag For Transmit Packets Contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits [15:13]: User priority • Bit 12: CFI or DEI • Bits [11:0]: MAC_VLAN_Tag_Data[VID]

72.18.26 MAC Q0 Tx Flow Control (MAC_Q0_Tx_Flow_Ctrl)

Offset

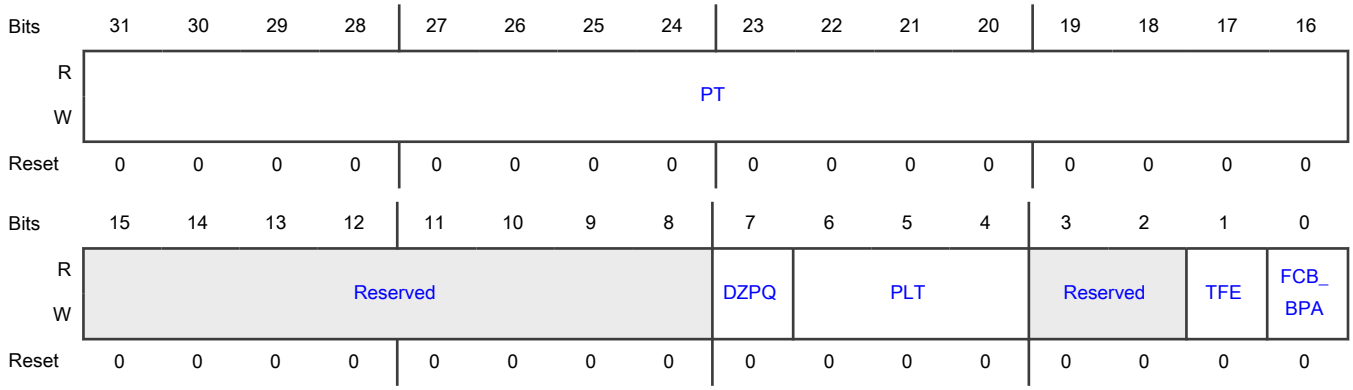
Register	Offset
MAC_Q0_Tx_Flow_Ctrl	70h

Function

Controls the generation and reception of the control (pause command) packets by MAC's flow control module. A write to the register with the Busy field = 1 triggers the flow control block to generate a pause packet. The 802.3x specification indicates the way the fields of the control packet are selected, and the pause time value from this register is used in the PT field of the control packet. The Busy field remains 1 until the control packet is transferred onto the cable. You must make sure that the Busy field is 0 before writing to this register.

If the value of the PFCE field in the MAC_Rx_Flow_Ctrl register is 1, this register controls the generation of priority flow control (PFC) frames with priorities mapped according to MAC_RxQ_Ctrl2[PSRQ0].

Diagram



Fields

Field	Function
31-16 PT	Pause Time Holds the value to be used in this field in the transmit control packet. If the bits of this field are configured to be double-synchronized with the (G)MII clock domain, consecutive writes to this register must be performed only after at least four clock cycles in the destination clock domain.
15-8 —	Reserved
7 DZPQ	Disable Zero-Quanta Pause Enables or disables zero-quanta pause packet generation. • If this field is 1, it disables the automatic generation of the zero-quanta pause packets that the flow-control signal from the FIFO layer (MTL or external sideband flow control signal sbd_flowctrl_i or mti_flowctrl_i) deasserts. • If this field is 0, normal operations with automatic zero-quanta pause packet generation are enabled. 0b - Enabled 1b - Disabled
6-4 PLT	Pause Low Threshold Configures the threshold of the pause timer at which the input flow control signal mti_flowctrl_i (or sbd_flowctrl_i) is checked for automatic retransmission of the pause packet. The threshold values must always be less than the pause time configured in bits [31:16]. For example, if the PT field = 100H (256 slot times), and the PLT field = 001, a second pause packet automatically transmits if the mti_flowctrl_i signal is asserted at 228 (256-28) slot times after the first pause packet transmits. This list provides the threshold values for different values: • The slot time is defined as the time taken to transmit 512 bits (64 bytes) on GMII or MII. • This (approximate) computation is based on the packet size (64, 1518, 2000, 9018, 16384, or 32768) + two pause packet sizes + IPG in slot times. 000b - Pause time minus 4 slot times (PT is 4 slot times) 001b - Pause time minus 28 slot times (PT is 28 slot times) 010b - Pause time minus 36 slot times (PT is 36 slot times) 011b - Pause time minus 144 slot times (PT is 144 slot times) 100b - Pause time minus 256 slot times (PT is 256 slot times) 101b - Pause time minus 512 slot times (PT is 512 slot times) 110b - Reserved
3-2 —	Reserved
1	Transmit Flow Control Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TFE	Indicates whether transmit flow control is enabled or disabled. • Full-Duplex mode: If this field is 1, MAC enables the flow control operation to the transmit pause packets. When the field becomes 0, MAC's flow control operation disables and MAC does not transmit any pause packets. • Half-Duplex mode: If this field is 1, MAC enables the backpressure operation. When the field becomes 0, the backpressure operation disables. 0b - Disabled 1b - Enabled
0 FCB_BPA	Flow Control Busy Or Backpressure Activate Initiates if a pause packet is in Full-Duplex mode and activates the backpressure function in Half-Duplex mode if the TFE field is 1. • Full-Duplex mode: In this mode, FCB_BPA must be read as 1'b0 before writing to this register. To initiate a pause packet, the application must set this field to 1'b1. During control packet transfer, the field's value continues to be 1 to indicate that a packet transmission is in progress. When pause packet transmission is complete, MAC resets this field to 1'b0. You must not write to this register until this field becomes 0. • Half-Duplex mode: If FCB_BPA = 1 and TFE = 1 too, MAC asserts backpressure in this mode. During the backpressure function, when MAC receives a new packet, the transmitter starts sending a JAM pattern resulting in a collision. This control register field is logically ORed with the mti_flowctrl_i input signal for the backpressure function. When you configure MAC using Full-Duplex mode, this field automatically becomes 0. Access restrictions apply to this field: writing 1 sets it, writing 0 has no effect, and the field is self-clearing. 0b - Flow control busy or backpressure activate is disabled 1b - Flow control busy or backpressure activate is enabled

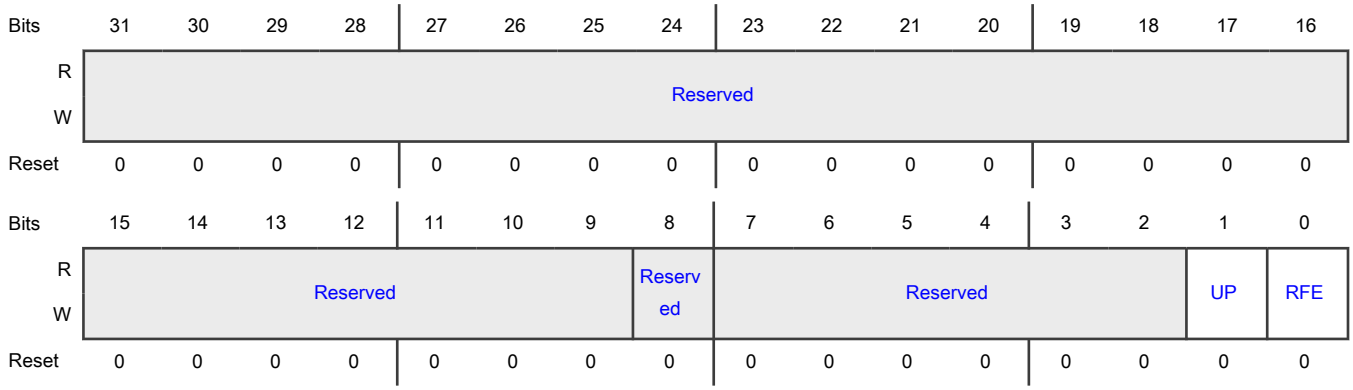
72.18.27 MAC Receive Flow Control (MAC_Rx_Flow_Ctrl)

Offset

Register	Offset
MAC_Rx_Flow_Ctrl	90h

Function
Controls the pausing of MAC transmit based on the received pause packet.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 —	Reserved
7-2 —	Reserved
1 UP	Unicast Pause Packet Detect Indicates whether the unicast pause packet is enabled or disabled. A pause packet is processed when it has the unique multicast address specified in IEEE 802.3. • If this field is 1, MAC can detect pause packets with a unicast address of the station. This address must comply with the specifications in MAC Address 0 High (MAC_Address0_High) and MAC Address 0 Low (MAC_Address0_Low). • If this field is 0, MAC only detects pause packets with a unique multicast address. **NOTE** MAC does not process a pause packet if the multicast address is different from the unique multicast address. This applies to the received PFC packet if PFC is enabled. The unique multicast address (01_80_C2_00_00_01h) complies with the IEEE 802.1 Qbb-2011 specifications. 0b - Disabled 1b - Enabled
0 RFE	Receive Flow Control Enable Indicates whether the receive flow control is enabled or disabled. • If this field is 1 and MAC is operating in Full-Duplex mode, MAC decodes the received pause packet and disables its transmitter for a specified (pause) time.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 0 or if MAC is operating in Half-Duplex mode, the decode function of the pause packet disables. When PFC is enabled, the flow control for PFC packets is enabled too. MAC decodes the received PFC packet and disables the transmit queue, with matching priorities, for a duration of the received pause time. 0b - Disabled 1b - Enabled

72.18.28 MAC RxQ Control 4 (MAC_RxQ_Ctrl4)

Offset

Register	Offset
MAC_RxQ_Ctrl4	94h

Function

Controls the routing of unicast and multicast packets that fail the destination or source address filter to the receive queues.

Diagram

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

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

Fields

Field	Function
31-18 —	Reserved
17 VFFQ	VLAN Tag Filter Fail Packets Queue Holds the receive queue number to which the tagged packets failing the destination or source address filter (and UFFQE and MFFQE not being 1) or failing the VLAN tag filter must be routed to. This field is valid only when the VFFQE field = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 VFFQE	VLAN Tag Filter Fail Packets Queuing Enable Indicates whether VLAN tag filter fail packet queuing is enabled or disabled. • If this field is 1, the tagged packets that fail the destination or source address filter or fail the VLAN tag filter are routed to the receive queue number programmed using the VFFQ field. • If this field is 0, the tagged packets that fail the destination or source address filter or fail the VLAN tag filter are routed based on the other routing options. This field is valid only when MAC_Packet_Filter[RA] = 1. 0b - Disabled 1b - Enabled
15-10 —	Reserved
9 MFFQ	Multicast Address Filter Fail Packets Queue Holds the receive queue number to which the multicast packets failing the destination or source address filter are routed to. This field is valid only when the MFFQE field = 1.
8 MFFQE	Multicast Address Filter Fail Packets Queuing Enable Indicates whether multicast address filter fail packet queuing is enabled or disabled. • If this field is 1, the multicast packets that fail the destination or source address filter are routed to the receive queue number programmed using the MFFQ field. • If this field is 0, the multicast packets that fail the destination or source address filter are routed based on the other routing options. This field is valid only when MAC_Packet_Filter[RA] = 1. 0b - Disabled 1b - Enabled
7-2 —	Reserved
1 UFFQ	Unicast Address Filter Fail Packets Queue Holds the receive queue number to which the unicast packets failing the destination or source address filter are routed to. This field is valid only when the UFFQE field = 1.
0 UFFQE	Unicast Address Filter Fail Packets Queuing Enable Indicates whether unicast address filter fail packet queuing is enabled or disabled. • When the value of this field is 1, the unicast packets that fail the destination or source address filter are routed to the receive queue number programmed using the UFFQ field. • If this field is 0, the unicast packets that fail the destination or source address filter are routed based on the other routing options.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is valid only when MAC_Packet_Filter[RA] = 1. 0b - Disabled 1b - Enabled

72.18.29 MAC RxQ Control 0 (MAC_RxQ_Ctrl0)

Offset

Register	Offset
MAC_RxQ_Ctrl0	A0h

Function

Controls queue management in the MAC receiver.

NOTE

In multiple receive queues configuration, all the queues are disabled by default. Enable the receive queue by programming the corresponding field in this register.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		RXQ1EN		RXQ0EN	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 RXQ1EN	Receive Queue 1 Enable Performs in a manner similar to the RXQ0EN field and indicates if receive queue 1 is enabled for AV or DCB. 00b - Queue not enabled 01b - Queue enabled for AV 10b - Queue enabled for DCB/generic 11b - Reserved
1-0 RXQ0EN	Receive Queue 0 Enable Indicates whether receive queue 0 is enabled for AV or DCB. 00b - Queue not enabled 01b - Queue enabled for AV 10b - Queue enabled for DCB/generic 11b - Reserved

72.18.30 Receive Queue Control 1 (MAC_RxQ_Ctrl1)

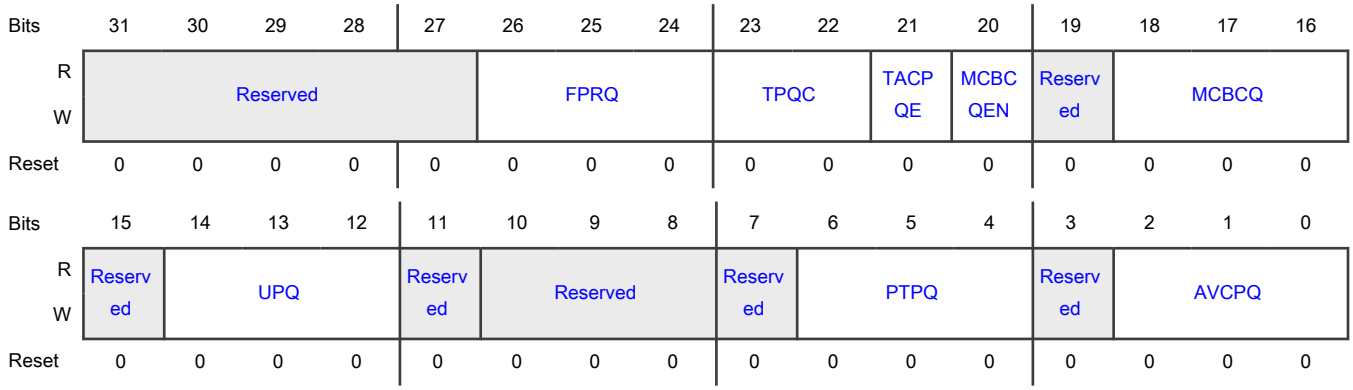
Offset

Register	Offset
MAC_RxQ_Ctrl1	A4h

Function

Controls the routing of multicast, broadcast, AV, DCB, and untagged packets to the receive queues.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 FPRQ	Frame Preemption Residue Queue Holds the receive queue number to which the residual preemption frames must be forwarded. Preemption frames that are tagged and pass the SA, DA, or VLAN filtering are routed based on the settings of the PSRQ fields in MAC RxQ Control 2 (MAC_RxQ_Ctrl2). All other frames are treated as residual frames and are routed to the queue number mentioned in this field. Queue 0 is used as the default queue for express frames, so you cannot write 0 to this field.
23-22 TPQC	Tagged PTP Over Ethernet Packets Queuing Control Controls the routing of the VLAN-tagged PTPoE packets. These are the allowed programmable options: • 2'b00: VLAN-tagged PTPoE packets are routed as generic VLAN-tagged packets (based on the PSRQ fields in MAC RxQ Control 2 (MAC_RxQ_Ctrl2) for only non-AV enabled receive queues). • 2'b01: VLAN-tagged PTPoE packets are routed to the receive queue that the PTPQ field of this register specifies (this Rx queue can be enabled for AV or non-AV traffic). • 2'b10: VLAN-tagged PTPoE packets are routed to only AV-enabled receive queues based on the settings of the PSRQ fields.
21 TACPQE	Tagged AV Control Packets Queuing Enable Indicates the status of tagged AV control packet queuing. • If this field is 1, MAC routes the received tagged AV control packets to the receive queue that the AVCPQ field specifies. • If this field is 0, MAC routes the received tagged AV control packets based on the tag priority matching the PSRQ fields in MAC RxQ Control 2 (MAC_RxQ_Ctrl2). 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 MCBCQEN	Multicast And Broadcast Queue Enable Specifies whether multicast or broadcast packet routing to the receive queue is enabled or disabled. The multicast or broadcast packets must be routed to the receive queue specified in the MCBCQ field. 0b - Disabled 1b - Enabled
19 —	Reserved
18-16 MCBCQ	Multicast And Broadcast Queue Specifies the receive queue onto which multicast or broadcast packets are routed. Any receive queue enabled for Generic, DCB, or AV traffic can be used to route the multicast or broadcast packets. 000b - Receive queue 0 001b - Receive queue 1 010b - Receive queue 2 011b - Receive queue 3 100b - Receive queue 4 101b - Receive queue 5 110b - Receive queue 6 111b - Receive queue 7
15 —	Reserved
14-12 UPQ	Untagged Packet Queue Indicates the receive queue to which the untagged packets need to be routed. Any receive queue enabled for generic, DCB, or AV traffic can be used to route the untagged packets. 000b - Receive queue 0 001b - Receive queue 1 010b - Receive queue 2 011b - Receive queue 3 100b - Receive queue 4 101b - Receive queue 5 110b - Receive queue 6 111b - Receive queue 7
11 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
10-8 —	Reserved
7 —	Reserved
6-4 PTPQ	PTP Packets Queue Specifies the receive queue on which the PTP packets sent over the Ethernet payload (not over IPv4 or IPv6) are routed. When MAC_Timestamp_Control[AV8021ASMEN] = 1, only untagged PTP over Ethernet packets are routed to a receive queue. If this field is not 1, then based on the settings of the TPQC field, both tagged and untagged PTPoE packets can be routed to this receive queue. 000b - Receive queue 0 001b - Receive queue 1 010b - Receive queue 2 011b - Receive queue 3 100b - Receive queue 4 101b - Receive queue 5 110b - Receive queue 6 111b - Receive queue 7
3 —	Reserved
2-0 AVCPQ	AV Untagged Control Packets Queue Specifies the receive queue on which the received AV tagged and untagged control packets are routed. The AV tagged (when the TACPQE field = 1) and untagged control packets are routed to the receive queue that this field specifies. 000b - Receive queue 0 001b - Receive queue 1 010b - Receive queue 2 011b - Receive queue 3 100b - Receive queue 4 101b - Receive queue 5 110b - Receive queue 6 111b - Receive queue 7

72.18.31 MAC RxQ Control 2 (MAC_RxQ_Ctrl2)

Offset

Register	Offset
MAC_RxQ_Ctrl2	A8h

Function

Controls the routing of tagged packets based on the settings of MAC_Ext_Configuration[USP] for packets received in receive queues 0-3.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PSRQ1								PSRQ0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 PSRQ1	Priorities Selected In Receive Queue 1 Determines the priorities assigned to receive queue 1. All the packets with priorities that match the values defined in this field are routed to receive queue 1. For example, if PSRQ1[4] = 1, packets with MAC_Ext_Configuration[USP] equal to 4 are routed to receive queue 1. You must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues. This means that the same priority is not mapped to multiple receive queues.
7-0 PSRQ0	Priorities Selected In Receive Queue 0 Determines the priorities assigned to receive queue 0. All the packets with priorities that match the values defined in this field are routed to receive queue 0. For example, if PSRQ0[5] = 1, packets with MAC_Ext_Configuration[USP] equal to 5 are routed to receive queue 0. You must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues. This means that the same priority is not mapped to multiple receive queues.

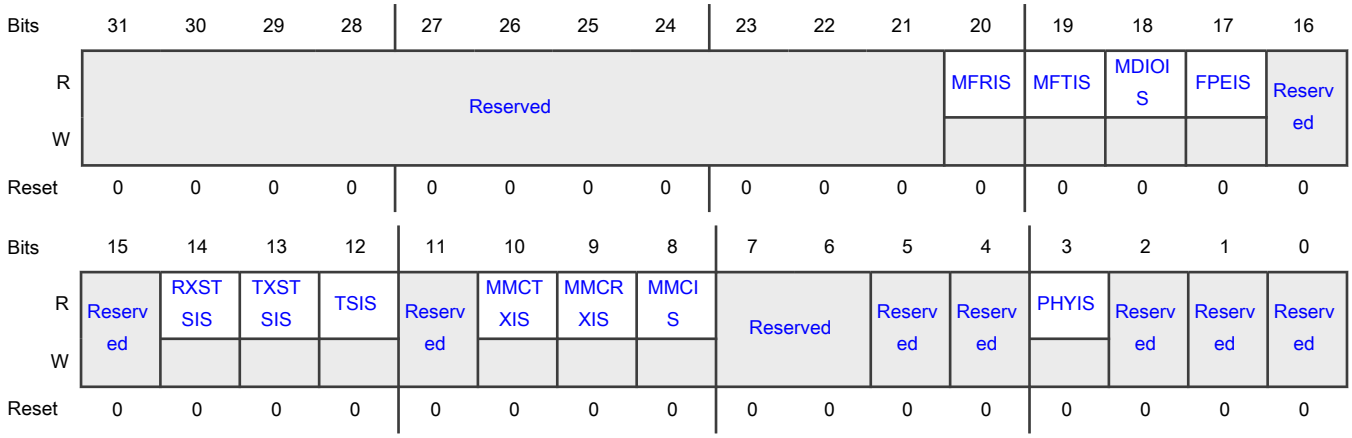
72.18.32 MAC Interrupt Status (MAC_Interrupt_Status)

Offset

Register	Offset
MAC_Interrupt_Status	B0h

Function
Contains the status of interrupts.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 MFRIS	MMC FPE Receive Interrupt Status Indicates the status of MMC FPE receive interrupt. This field becomes 1 when any of the fields in MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt) is 1. The field becomes 0 when all the other fields in MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt) become 0 too. 0b - Inactive 1b - Active
19 MFTIS	MMC FPE Transmit Interrupt Status Indicates the status of MMC FPE transmit interrupt. This field becomes 1 when an interrupt generates in MMC Transmit FPE Fragment Counter Interrupt Status (MMC_FPE_Tx_Interrupt). The field becomes 0 when all the other fields in this register become 0 too. This field is valid only when you select the Enable MAC Management Counters (MMC) option along with FPE support.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Inactive 1b - Active
18 MDIOIS	MDIO Interrupt Status Indicates the status of an interrupt event after the completion of an MDIO operation. To reset this field, you must read this field or write 1 to it when MAC_CSR_SW_Ctrl[RCWE] = 1. Access restrictions apply to this field. It clears on a read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. The field automatically becomes 1 on an internal event. 0b - Inactive 1b - Active
17 FPEIS	Frame Preemption Interrupt Status Indicates the status of an interrupt event during the frame preemption operation (the value of RVER, RRSP, TVER, and TRSP fields of MAC FPE Control STS (MAC_FPE_CTRL_STS) = 1). To reset this field, you must clear the event in MAC FPE Control STS (MAC_FPE_CTRL_STS) that caused the interrupt. 0b - Inactive 1b - Ative
16 —	Reserved
15 —	Reserved
14 RXSTSIS	Receive Status Interrupt Indicates the status of received packets. This field becomes 1 when MAC_Rx_Tx_Status[RWT] = 1. The field becomes 0 when you read the corresponding interrupt source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status), or if you write to this interrupt source field when MAC_CSR_SW_Ctrl[RCWE] = 1. The field becomes 0 in either of these circumstances: • You read the corresponding interrupt-source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status). • You write 1 to the corresponding interrupt-source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status) when MAC_CSR_SW_Ctrl[RCWE] = 1. 0b - Inactive 1b - Active
13 TXSTSIS	Transmit Status Interrupt Indicates the status of transmitted packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 1 when any of the following fields is 1 in MAC Rx Transmit Status (MAC_Rx_Tx_Status): • Excessive collision (EXCOL) • Late collision (LCOL) • Excessive deferral (EXDEF) • Loss of carrier (LCARR) • No carrier (NCARR) • Jabber timeout (TJT) This field becomes 0 in either of these circumstances: • You read the corresponding interrupt-source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status). • You write 1 to the corresponding interrupt-source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status) when MAC_CSR_SW_Ctrl[RCWE] = 1. 0b - Inactive 1b - Active
12 TSIS	Timestamp Interrupt Status Indicates the status of timestamp interrupt. If the timestamp feature is enabled, this field becomes 1 when any of the following conditions is true: • The system time value is equal to or exceeds the value specified in MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds) and MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds). • There is an overflow in MAC System Time In Seconds (MAC_System_Time_Seconds). • The target time error occurred, which means, the programmed target time already elapsed. In configurations other than EQOS_CORE, when the drop transmit status is enabled in MTL, this field becomes 1 when the captured transmit timestamp is updated in MAC Transmit Timestamp Status In Nanoseconds (MAC_Tx_Timestamp_Status_Nanoseconds) and MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds). The field becomes 0 in either of these circumstances: • You read the corresponding interrupt-source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status). • You write 1 to the corresponding interrupt-source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status) when MAC_CSR_SW_Ctrl[RCWE] = 1. 0b - Inactive 1b - Active
11 —	Reserved
10 MMCTXIS	MMC Transmit Interrupt Status Indicates the status of MMC transmit interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field: • Becomes 1 when any field in MMC Transmit Interrupt (MMC_Tx_Interrupt) becomes 1. • Becomes 0 when all the fields in MMC Transmit Interrupt (MMC_Tx_Interrupt)return to 0. This field is valid only when you select the Enable MAC Management Counters (MMC) option. 0b - Inactive 1b - Active
9 MMCRXIS	MMC Receive Interrupt Status Indicates the status of MMC receive interrupt. This field: • Becomes 1 when any field in MMC Receive Interrupt (MMC_Rx_Interrupt) becomes 1. • Becomes 0 when all the fields in MMC Receive Interrupt (MMC_Rx_Interrupt)return to 0. This field is valid only when you select the Enable MAC Management Counters (MMC) option. 0b - Inactive 1b - Active
8 MMCIS	MMC Interrupt Status Indicates the status of MMC interrupt. This field becomes 1 when MMCRXIS = MMCTXIS = 1, and the field becomes 0 only when MMCRXIS = MMCTXIS = 0. MMCIS is valid only when you select the Enable MAC Management Counters (MMC) option. 0b - Inactive 1b - Active
7-6 —	Reserved
5 —	Reserved
4 —	Reserved
3 PHYIS	PHY Interrupt Indicates whether the PHY interrupt is detected. This field becomes 1 when rising edge is detected on the phy_intr_i input signal. The field becomes 0 when you read this register (or if you write 1 to this field when MAC_CSR_SW_Ctrl[RCWE] = 1). 0b - Not detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Detected
2 —	Reserved
1 —	Reserved
0 —	Reserved

72.18.33 MAC Interrupt Enable (MAC_Interrupt_Enable)

Offset

Register	Offset
MAC_Interrupt_Enable	B4h

Function

Contains masks for generating MAC interrupts.

Diagram

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

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

Fields

Field	Function
31-19 —	Reserved
18	MDIO Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
MDIOIE	Indicates whether the MDIO interrupt is enabled or disabled. If this field is 1, it enables the assertion of the interrupt when MAC_Interrupt_Status[MDIOIS] = 1. 0b - Disabled 1b - Enabled
17 FPEIE	Frame Preemption Interrupt Enable Indicates whether the frame preemption interrupt is enabled or disabled. If this field is 1, it enables the assertion of the interrupt when MAC_Interrupt_Status[FPEIS] = 1. 0b - Disabled 1b - Enabled
16 —	Reserved
15 —	Reserved
14 RXSTSIE	Receive Status Interrupt Enable Indicates whether the receive status interrupt is enabled or disabled. If this field is 1, it enables the assertion of the interrupt signal because of the setting of MAC_Interrupt_Status[RXSTSIS]. 0b - Disabled 1b - Enabled
13 TXSTSIE	Transmit Status Interrupt Enable Indicates whether the timestamp status interrupt is enabled or disabled. If this field is 1, it enables the assertion of the interrupt signal because of the setting of MAC_Interrupt_Status[TXSTSIS]. 0b - Disabled 1b - Enabled
12 TSIE	Timestamp Interrupt Enable Indicates whether the timestamp interrupt is enabled or disabled. If this field is 1, it enables the assertion of the interrupt signal because of the setting MAC_Interrupt_Status[TSIS]. 0b - Disabled 1b - Enabled
11-6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5 —	Reserved
4 —	Reserved
3 PHYIE	PHY Interrupt Enable Indicates whether the assertion of the interrupt signal is enabled or disabled. If this field is 1, it enables the assertion of the interrupt signal because of the setting of MAC_Interrupt_Status[PHYIS]. 0b - Disabled 1b - Enabled
2 —	Reserved
1 —	Reserved
0 —	Reserved

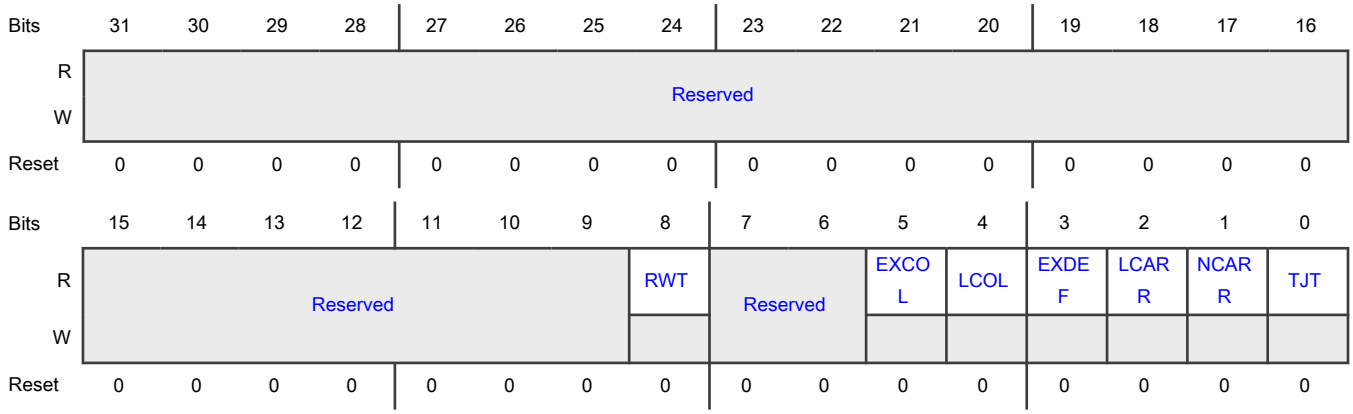
72.18.34 MAC Rx Transmit Status (MAC_Rx_Tx_Status)

Offset

Register	Offset
MAC_Rx_Tx_Status	B8h

Function
Contains the receive and transmit error status.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 RWT	Receive Watchdog Timeout Indicates the status of receive watchdog. This field becomes 1 when a packet with a length greater than 2,048 bytes is received (10,240 bytes when Jumbo Packet mode is enabled) and MAC_Configuration[WD] = 0. RWT becomes 1 when a packet with length greater than 16,383 bytes is received and MAC_Configuration[WD] = 1. Access restrictions apply to RWT. It becomes 0 on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. RWT automatically becomes 1 on an internal event occurrence. 0b - No receive watchdog timed out 1b - Receive watchdog timed out
7-6 —	Reserved
5 EXCOL	Excessive Collisions Indicates the status of excessive collision. If MTL_Operation_Mode[DTXSTS] = 1, this field, EXCOL, indicates that the transmission aborted after 16 successive collisions while attempting to transmit the current packet. If MAC_Configuration[DR] = 1, EXCOL becomes 1 after the first collision and the packet transmission is aborted. Access restrictions apply to EXCOL. It becomes 0 on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. EXCOL automatically becomes 1 on an internal event occurrence. 0b - No collision 1b - Excessive collision is sensed
4	Late Collision

Table continues on the next page...

Table continued from the previous page...

Field	Function
LCOL	Indicates the status of late collision. If MTL_Operation_Mode[DTXSTS] = 1, this field, LCOL, indicates that the packet transmission aborted because a collision occurred after the collision window (64 bytes including preamble in MII mode; 512 bytes including preamble and carrier extension in GMII mode). This field is invalid if an underflow error occurs. Access restrictions apply to LCOL. It becomes 0 on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. LCOL automatically becomes 1 on an internal event occurrence. 0b - No collision 1b - Late collision is sensed
3 EXDEF	Excessive Deferral Indicates whether the transmission ended because of excessive deferral. If MTL_Operation_Mode[DTXSTS] = 1 and MAC_Configuration[DC] = 1 too, this field, EXDEF, indicates that the transmission ended because of excessive deferral of over 24,288 bit times (155,680 in 1000/2500 Mbit/s mode or when the jumbo packet is enabled). Access restrictions apply to EXDEF. It becomes 0 on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. EXDEF automatically becomes 1 on an internal event occurrence. 0b - No excessive deferral 1b - Excessive deferral
2 LCARR	Loss of Carrier Indicates the status of carrier signal. If MTL_Operation_Mode[DTXSTS] = 1, this field, LCARR, indicates that the loss of carrier occurred during packet transmission, which means, the phy_crs_i signal was inactive for one or more transmission clock periods during packet transmission. The field is valid only for packets transmitted without collision. Access restrictions apply to LCARR. It becomes 0 on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. LCARR automatically becomes 1 on an internal event occurrence. 0b - Carrier is present 1b - Loss of carrier
1 NCARR	No Carrier Indicates whether the carrier signal is present from the PHY at the end of preamble transmission. If MTL_Operation_Mode[DTXSTS] = 1, NCARR indicates that the carrier signal from the PHY is absent at the end of preamble transmission. Access restrictions apply to NCARR. It becomes 0 after either of these actions: • You read NCARR. • You write 1 to NCARR when MAC_CSR_SW_Ctrl[RCWE] = 1. NCARR automatically becomes 1 after an internal event occurs.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Present 1b - Absent
0 TJT	Transmit Jabber Timeout Indicates whether a transmit-jabber timer timeout occurred. The timeout occurs after any of these events: • MAC_Configuration[JD] = 0 and the normal packet size exceeds 2,048 bytes. • MAC_Configuration[JD] = 0 and the jumbo packet size exceeds 10240 bytes. • MAC_Configuration[JD] = 1 and the normal packet size exceeds 16383 bytes. Access restrictions apply to TJT. It becomes 0 after either of these actions: • You read TJT. • You write 1 to TJT when MAC_CSR_SW_Ctrl[RCWE] = 1. TJT automatically becomes 1 after an internal event occurs. 0b - No transmit jabber timeout occurred 1b - Transmit jabber timeout occurred

72.18.35 MAC Version (MAC_Version)

Offset

Register	Offset
MAC_Version	110h

Function

Identifies the version of the module.

Diagram

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

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

Fields

Field	Function
31-16 —	Reserved
15-8 CFGVER	IP Configuration Version
7-0 IPVER	IP Version

72.18.36 MAC Debug (MAC_Debug)

Offset

Register	Offset
MAC_Debug	114h

Function
Provides the debug status of the various MAC blocks.

Diagram

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

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

Fields

Field	Function
31-19 —	Reserved
18-17	MAC Transmit Packet Controller Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
TFCSTS	Indicates the state of the MAC transmit packet controller module. 00b - Idle state 01b - Waiting for one of these: status of the previous packet or IPG, or for the back-off period to be over 10b - Generating and transmitting a pause control packet (in Full-Duplex mode) 11b - Transferring input packet for transmission
16 TPESTS	MAC GMII Or MII Transmit Protocol Engine Status Indicates whether MAC GMII or MII transmit protocol engine is detected. If this field is 1, it indicates that the MAC GMII or MII transmit protocol engine is actively transmitting data and is not in an idle state. 0b - Not detected 1b - Detected
15-3 —	Reserved
2-1 RFCFCSTS	MAC Receive Packet Controller FIFO Status Indicates the status of the small FIFO read and write controllers of the MAC receive packet controller module. If this field is 1, it indicates that the small FIFO read and write controllers of the MAC receive packet controller module are in an active state. 00b - Inactive 01b - Active
0 RPESTS	Receive Protocol Engine Status Indicates whether MAC GMII or MII receive protocol engine is detected. If this field is 1, it indicates that the MAC GMII or MII receive protocol engine is actively receiving data and is not in an idle state. 0b - Not detected 1b - Detected

72.18.37 MAC Hardware Feature 0 (MAC_HW_Feature0)

Offset

Register	Offset
MAC_HW_Feature0	11Ch

Function

Indicates the presence of the first set of EMAC optional features or functions. You can use this register to dynamically enable or disable the programs related to these features.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	ACTPHYSEL			SAVLANI...	TSSTSSEL		MACADR6...	MACADR3...	ADDMACADRSEL					Reserved	RXCOESEL
W																
Reset	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	TXCOESEL	EEESEL	TSSEL	Reserved		ARPOFFS...	MMCSEL	MGKSEL	RWKSEL	SMASEL	VLHASH	PCSSEL	HDSEL	GMIISEL	MIISEL
W																
Reset	0	1	0	1	0	0	0	1	0	0	1	1	0	1	0	1

Fields

Field	Function
31 —	Reserved
30-28 ACTPHYSEL	Active PHY Feature Indicates the selected active PHY interface. If you have multiple PHY interfaces (GMII, MII, RevMII, RGMII, RMII, RTBI, SGMII, SGMII, or TBI) in your configuration, this field indicates the sampled value of the phy_intf_sel_i signal during reset deassertion. 000b - GMII or MII 001b - RGMII 010b - SGMII 011b - TBI 100b - RMII 101b - RTBI 110b - SMII 111b - RevMII
27 SAVLANINS	SA or VLAN Insertion Feature Indicates whether the feature that enables SA or VLAN insertion on transmit is selected. This field becomes 1 if you select the "enable SA and VLAN insertion on Tx" option. 0b - Not selected 1b - Selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
26-25 TSTSSEL	Timestamp System Time Source Feature Indicates the source of the timestamp system time. This field becomes 1 if you select the "enable IEEE 1588 timestamp support" option. 00b - Internal 01b - External 10b - Both internal and external 11b - Reserved
24 MACADR64SEL	MAC Addresses 64-127 Indicates whether the feature that enables additional 64 MAC address regsiters (64-127) is selected. This field becomes 1 if you enable the "MAC addresses (64-127) select" option. 0b - Not selected 1b - Selected
23 MACADR32SEL	MAC Addresses 32-63 Indicates whether the feature that enables additional 32 MAC address registers (32-63) is selected. This field becomes 1 if you select the "MAC addresses 32-63 select" option. 0b - Not selected 1b - Selected
22-18 ADDMACADRSEL	MAC Addresses 1-31 Indicates whether the feature that enables additional 1-31 MAC address registers is selected. This field becomes 1 if you select a non-zero value for enabling the "MAC addresses 1-31 select" option. 0_0000b - Not selected 0_0001b - Selected
17 —	Reserved
16 RXCOESEL	Receive Checksum Offload Feature Indicates whether the feature that enables receive TCP/IP checksum offload is selected. This field becomes 1 if you select the "enable receive TCP/IP checksum check" option. 0b - Not selected 1b - Selected
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 TXCOESEL	Transmit Checksum Offload Feature Indicates whether the feature that enables transmit TCP/IP checksum insertion is selected. This field becomes 1 if you select the "enable transmit TCP/IP checksum insertion" option. 0b - Not selected 1b - Selected
13 EEESEL	Energy Efficient Ethernet (EEE) Feature Indicates whether the feature that enables EEE is selected. This field becomes 1 if you select the "enable EEE" option. 0b - Not selected 1b - Selected
12 TSSEL	IEEE 1588-2008 Timestamp Feature Indicates whether the option that enables IEEE 1588-2008 timestamp is selected. This field becomes 1 if you select the "enable IEEE 1588 timestamp support" option. 0b - Not selected 1b - Selected
11-10 —	Reserved
9 ARPOFFSEL	ARP Offload Feature Indicates whether the feature that enables ARP offload is selected. This field becomes 1 if you select the "enable IPv4 ARP offload" option. 0b - Not selected 1b - Selected
8 MMCSEL	MAC Management Counters (MMC) Feature Indicates whether the feature that enables MMC is selected. This field becomes 1 if you select the "enable MMC" option. 0b - Not selected 1b - Selected
7 MGKSEL	PMT Magic Packet Feature Indicates whether the feature that enables PMT magic detection is selected. This field becomes 1 if you select the "enable magic packet detection" option. 0b - Not selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Selected
6 RWKSEL	PMT Remote Wake-Up Packet Feature Indicates whether the feature that enables PMT remote wake-up packet detection is selected. This field becomes 1 if you select the "enable remote wake-up packet detection" option. 0b - Not selected 1b - Selected
5 SMASEL	SMA (MDIO) Interface Feature Indicates whether the feature that enables station management (MDIO interface) is selected. This field becomes 1 if you select the "enable station management (MDIO interface)" option. 0b - Not selected 1b - Selected
4 VLHASH	VLAN Hash Filter Feature Indicates whether the filtering option based on VLAN hash table is selected. This field becomes 1 if you select the "enable VLAN hash table based filtering" option. 0b - Not selected 1b - Selected
3 PCSSEL	PCS Select Indicates if either of the TBI, SGMII, or RTBI PHY interface options is selected. This field becomes 1 if you select any one of the TBI, SGMII, or RTBI PHY interface options. 0b - No 1b - Yes
2 HDSEL	Half-Duplex Support Feature Indicates whether half-duplex support is available as the mode of operation. This field becomes 1 if you select Half-Duplex as the mode of operation. 0b - Unavailable 1b - Available
1 GMIISEL	1000 Mbit/s Support Feature Indicates whether 1000 Mbit/s support is available as the mode of operation. This field becomes 1 if you select 1000 Mbit/s as the mode of operation. 0b - Unavailable 1b - Available

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 MIISEL	10 or 100 Mbit/s Support Feature Indicates whether 10 or 100 Mbit/s support is available as the mode of operation. This field becomes 1 if you select 10/100 Mbit/s as the mode of operation. 0b - Unavailable 1b - Available

72.18.38 MAC Hardware Feature 1 (MAC_HW_Feature1)

Offset

Register	Offset
MAC_HW_Feature1	120h

Function

Indicates the presence of the second set of optional features or functions of the module. You can use this register to dynamically enable or disable the programs related to these features.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	L3L4FNUM			Reserved	HASHTBLSZ			POUOST	Reserved	RAVSEL	AVSEL	DBGMEMA	TSOEN	SPHEN	DCBEN
W																
Reset	0	0	1	0	0	0	0	1	0	0	0	1	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR64		ADVTHWO...	PTOEN	OSTEN	TXFIFOSIZE					SPRAM	RXFIFOSIZE				
W																
Reset	0	0	1	0	1	0	0	1	1	0	1	0	0	1	1	0

Fields

Field	Function
31 —	Reserved
30-27	L3 Or L4 Filter Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
L3L4FNUM	Indicates the total number of L3 or L4 filters. 0000b - No filters (0) 0001b - 1 0010b - 2 0011b - 3 0100b - 4 0101b - 5 0110b - 6 0111b - 7 1000b - 8
26 —	Reserved
25-24 HASHTBLSZ	Hash Table Size Indicates the size of the hash table. 00b - No hash table 01b - 64 10b - 128 11b - 256
23 POUOST	One Step For PTP Over UDP/IP Feature Indicates whether the feature that enables one step for PTP over UDP/IP is selected. This field becomes 1 if you select the "enable one-step timestamp for PTP over UDP/IP" option. 0b - Not selected 1b - Selected
22 —	Reserved
21 RAVSEL	Receive Side-Only AV Feature Indicates whether the feature that enables audio-video bridging only on the receive side is selected. This field becomes 1 if you select the "enable audio-video bridging on receive side only" option. 0b - Not selected 1b - Selected
20	AV Feature Indicates whether the feature that enables audio-video bridging is selected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
AVSEL	This field becomes 1 if you select the "enable audio video bridging" option. 0b - Not selected 1b - Selected
19 DBGMEMA	DMA Debug Registers Enable Feature Indicates whether the feature that enables DMA debug regsiters is selected. This field becomes 1 if you select the "enable DMA debug mode" option. 0b - Not selected 1b - Selected
18 TSOEN	TCP Segmentation Offload Enable Feature Indicates whether the feature that enables TCP segmentation offload for TCP/IP packets is selected. This field becomes 1 if you select the "enable TCP segmentation offloading for TCP/IP packets" option. 0b - Not selected 1b - Selected
17 SPHEN	Split Header Enable Feature Indicates whether the feature that enables a split header structure is selected. This field becomes 1 if you select the "enable split header structure" option. 0b - Not selected 1b - Selected
16 DCBEN	DCB Enable Feature Indicates whether the feature that enables data center bridging is selected. This field becomes 1 if you select the "enable data center bridging" option. 0b - Not selected 1b - Selected
15-14 ADDR64	Address Width Feature Indicates the configured address width. 00b - 32 01b - 40 10b - 48 11b - Reserved
13 ADVTHWORD	IEEE 1588 High-Word Feature Indicates whether the feature that enables the IEEE 1588 high-word register is selected. This field becomes 1 if you select the "enable/add IEEE 1588 higher-word register" option.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not selected 1b - Selected
12 PTOEN	PTP Offload Enable Feature Indicates whether the feature that enables PTP offload is selected. This field becomes 1 if you select the "enable PTP timestamp offload" option. 0b - Not selected 1b - Selected
11 OSTEN	One-Step Timestamping Enable Feature Indicates whether the feature that enables one-step timestamping is selected. This field becomes 1 if you select the "enable one-step timestamp" option. 0b - Not selected 1b - Selected
10-6 TXFIFOSIZE	MTL Transmit FIFO Size Feature Contains the configured value of MTL transmit FIFO (in bytes) expressed as log to base 2 minus 7, which means, Log 2 (TXFIFO_SIZE) - 7. 0_0000b - 128 bytes 0_0001b - 256 bytes 0_0010b - 512 bytes 0_0011b - 1024 bytes 0_0100b - 2048 bytes 0_0101b - 4096 bytes 0_0110b - 8192 bytes 0_0111b - 16384 bytes 0_1000b - 32 KB 0_1001b - 64 KB 0_1010b - 128 KB 0_1011b - Reserved
5 SPRAM	Single Port RAM Feature Indicates whether the single-port RAM feature is selected. This field becomes 1 if you select the option that enables the "use single port RAM" feature. 0b - Not selected 1b - Selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 RXFIFOSIZE	MTL Receive FIFO Size Feature Contains the configured value of MTL receive FIFO (in bytes) expressed as log to base 2 minus 7, which means, Log 2 (RXFIFO_SIZE) - 7. 0_0000b - 128 bytes 0_0001b - 256 bytes 0_0010b - 512 bytes 0_0011b - 1024 bytes 0_0100b - 2048 bytes 0_0101b - 4096 bytes 0_0110b - 8192 bytes 0_0111b - 16384 bytes 0_1000b - 32 KB 0_1001b - 64 KB 0_1010b - 128 KB 0_1011b - 256 KB 0_1100b - Reserved

72.18.39 MAC Hardware Feature 2 (MAC_HW_Feature2)

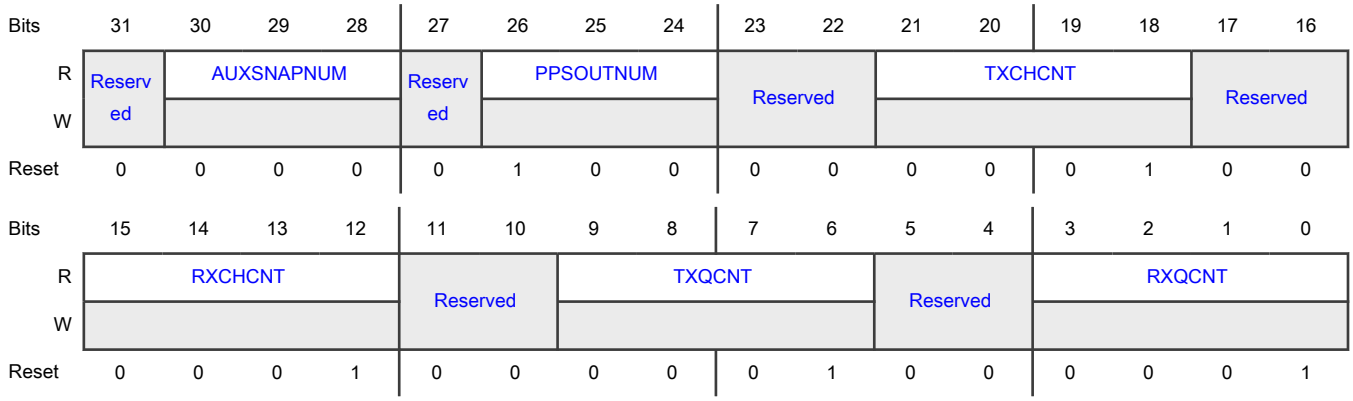
Offset

Register	Offset
MAC_HW_Feature2	124h

Function

Indicates the presence of the third set of optional features or functions of the module. You can use this register to dynamically enable or disable the programs related to these features.

Diagram



Fields

Field	Function
31 —	Reserved
30-28 AUXSNAPNUM	Number Of Auxiliary Snapshot Inputs Indicates the number of auxiliary snapshot inputs. 000b - No auxiliary input (0) 001b - 1 010b - 2 011b - 3 100b - 4 101b - Reserved
27 —	Reserved
26-24 PPSOUTNUM	Number Of PPS Outputs Indicates the number of PPS outputs. 000b - No PPS output (0) 001b - 1 010b - 2 011b - 3 100b - 4 101b - Reserved
23-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-18 TXCHCNT	Number Of DMA Transmit Channels Indicates the number of DMA transmit channels. 0000b - 1 0001b - 2 0010b - 3 0011b - 4 0100b - 5 0101b - 6 0110b - 7 0111b - 8
17-16 —	Reserved
15-12 RXCHCNT	Number Of DMA Receive Channels Indicates the number of DMA receive channels. 0000b - 1 0001b - 2 0010b - 3 0011b - 4 0100b - 5 0101b - 6 0110b - 7 0111b - 8
11-10 —	Reserved
9-6 TXQCNT	Number Of MTL Transmit Queues Indicates the number of MTL transmit queues. 0000b - 1 0001b - 2 0010b - 3 0011b - 4 0100b - 5 0101b - 6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0110b - 7 0111b - 8
5-4 —	Reserved
3-0 RXQCNT	Number Of MTL Receive Queues Indicates the number of MTL receive queues. 0000b - 1 0001b - 2 0010b - 3 0011b - 4 0100b - 5 0101b - 6 0110b - 7 0111b - 8

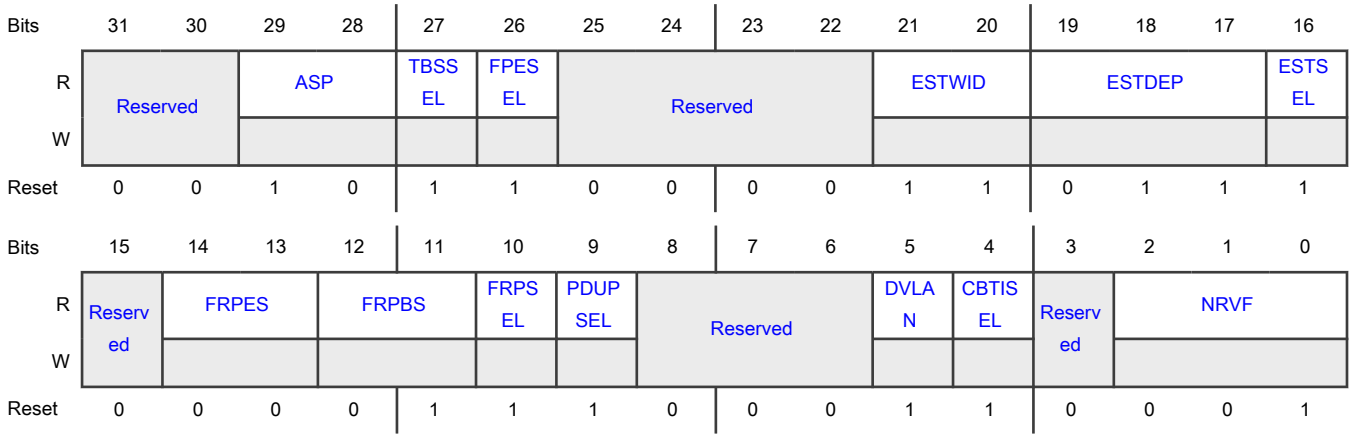
72.18.40 MAC Hardware Feature 3 (MAC_HW_Feature3)

Offset

Register	Offset
MAC_HW_Feature3	128h

Function
Indicates the presence of the fourth set of optional features or functions of the module. You can use this register to dynamically enable or disable the programs related to these features.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-28 ASP	Automotive Safety Package Indicates the encoding for the different automotive safety features. 00b - No safety features selected 01b - Only "ECC protection for external memory" feature is selected 10b - All the safety features are selected without the "parity port enable for external interface" feature 11b - All the safety features are selected with the "parity port enable for external interface" feature
27 TBSSEL	Time-Based Scheduling Feature Indicates whether the time-based scheduling feature is selected. This field becomes 1 if you select the option that enables the time-based scheduling feature. 0b - Selected 1b - Selected
26 FPESEL	Frame Preemption Feature Indicates whether the feature that enables frame preemption is selected. This field becomes 1 if you select the "enable frame preemption" option. 0b - Not selected 1b - Selected
25-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-20 ESTWID	Estimated Time Interval Width Indicates the width configured for the time interval field in the gate control list. 00b - Width not configured 01b - 16 bits 10b - 20 bits 11b - 24 bits
19-17 ESTDEP	Depth Of Gate Control List Indicates the depth of gate control list expressed as Log 2 (DWC_EQOS_EST_DEP) - 5. 000b - No depth configured 001b - 64 bytes 010b - 128 bytes 011b - 256 bytes 100b - 512 bytes 101b - 1024 bytes 110b - Reserved
16 ESTSEL	Enhancements To Scheduling Traffic Feature Indicates whether the "enhancements to scheduling traffic" feature is selected. This field becomes 1 if you select the "enable enhancements to scheduling traffic" option. 0b - Not selected 1b - Selected
15 —	Reserved
14-13 FRPES	Flexible Receive Parser Table Entry Size Indicates the maximum number of parser entries that the flexible receive parser supports. 00b - 64 01b - 128 10b - 256 11b - Reserved
12-11 FRPBS	Flexible Receive Parser Buffer Size Indicates the maximum number of packet data bytes that the flexible receive parser supports. 00b - 64

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - 128 10b - 256 11b - Reserved
10 FRPSEL	Flexible Receive Parser Feature Indicates whether the flexible receive parser feature is selected. This field becomes 1 if you select the "enable flexible programmable receive parser" option. 0b - Not selected 1b - Selected
9 PDUPSEL	Broadcast/Multicast Packet Duplication Feature Indicates whether the broadcast or multicast duplication feature is selected. This field becomes 1 if you select the "broadcast or multicast packet duplication" option. 0b - Not selected 1b - Selected
8-6 —	Reserved
5 DVLAN	Double VLAN Tag Processing Feature Indicates whether the double VLAN processing feature is selected. 0b - Not selected 1b - Selected
4 CBTISEL	Queue/Channel Based VLAN Tag Insertion On Transmit Feature Indicates whether the "queue- or channel-based VLAN tag insertion on Tx" feature is selected. This field becomes 1 if you select the "enable queue- or channel-based VLAN tag insertion on Tx" option. 0b - Not selected 1b - Selected
3 —	Reserved
2-0 NRVF	Number Of Extended VLAN Tag Filters Indicates the number of selected extended VLAN tag filters. 000b - No filters (0) 001b - 4 010b - 8

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - 16 100b - 24 101b - 32 110b - Reserved

72.18.41 MAC DPP FSM Interrupt Status (MAC_DPP_FSM_Interrupt_Status)

Offset

Register	Offset
MAC_DPP_FSM_Interrupt_Status	140h

Function

Contains the status of automative-safety related data path parity errors, interface timeout errors, FSM state parity errors, and FSM state timeout errors.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved	PTES	ATES	Reserved	RTES	TTES	Reserved	Reserved	ARPES	MTSPES	MPES	RDPES	Reserved	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved
24 FSMPES	FSM State Parity Error Status Indicates whether a parity error is detected on one of the FSM state registers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
23-18 —	Reserved
17 —	Reserved
16 MSTTES	Master Read Or Write Timeout Error Status Indicates whether an application or CSR timeout error is detected on the master (AXI, AHB, ARI, or ATI) interface. After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 PTES	PTP FSM Timeout Error Status Indicates whether one of the PTP FSM timeout errors is detected. After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
11 ATES	APP FSM Timeout Error Status Indicates whether one of the APP FSM timeout errors is detected. After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 RTES	Receive FSM Timeout Error Status Indicates whether one of the receive FSM timeout errors is detected. After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
8 TTES	Transmit FSM Timeout Error Status Indicates whether one of the transmit FSM timeout errors is detected. After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
7 —	Reserved
6 —	Reserved
5 ARPES	Application Receive Interface Data Path Parity Error Status Indicates whether the parity checker detected a parity error on the application receive interface (or on PC6 as shown in the transmit data path parity protection diagram). After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
4 MTSPES	MTL Transmit Status Data Path Parity Checker Error Status Indicates whether the parity checker detected a parity error for the MTL transmit status data on ATI (or on PC5 as shown in the transmit data path parity protection diagram). After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
3 MPES	MTL Data Path Parity Checker Error Status Indicates whether the parity checker detected a parity error on the MTL transmit write controller interface (or on PC4 as shown in the transmit data path parity protection diagram). After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 RDPES	Read Descriptor Parity Checker Error Status Indicates whether the parity checker detected a parity error on the DMA read descriptor interface (or on PC3 as shown in the transmit data path parity protection diagram). After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
1 —	Reserved
0 —	Reserved

72.18.42 MAC FSM Control (MAC_FSM_Control)

Offset

Register	Offset
MAC_FSM_Control	148h

Function

Is used to control the FSM state parity and timeout error injection in Debug mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	Reserved	Reserved	PLGRNML	ALGRNML	Reserved	RLGRNML	TLGRNML	Reserved	Reserved	Reserved	PPEIN	APEIN	Reserved	RPEIN	TPEIN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	Reserved	PTEIN	ATEIN	Reserved	RTEIN	TTEIN	Reserved						PRTYEN	TMOUTEN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
30 —	Reserved
29 —	Reserved
28 PLGRNML	PTP Large Or Normal Mode Select Indicates which mode of tic generation is used for the PTP domain. 0b - Normal mode 1b - Large mode
27 ALGRNML	APP Large Or Normal Mode Select Indicates which mode of tic generation is used for the APP domain. 0b - Normal mode 1b - Large mode
26 —	Reserved
25 RLGRNML	Receive Large Or Normal Mode Select Indicates which mode of tic generation is used for the receive domain. 0b - Normal mode 1b - Large mode
24 TLGRNML	Transmit Large Or Normal Mode Select Indicates which mode of tic generation is used for the transmit domain. 0b - Normal mode 1b - Large mode
23 —	Reserved
22 —	Reserved
21 —	Reserved
20	PTP FSM Parity Error Injection

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPEIN	Indicates the status of error injection for PTP FSM parity. 0b - Disabled 1b - Enabled
19 APEIN	APP FSM Parity Error Injection Indicates the status of error injection for APP FSM parity. 0b - Disabled 1b - Enabled
18 —	Reserved
17 RPEIN	Receive FSM Parity Error Injection Indicates the status of error injection for receive FSM parity. 0b - Disabled 1b - Enabled
16 TPEIN	Transmit FSM Parity Error Injection Indicates the status of error injection for transmit FSM parity. 0b - Disabled 1b - Enabled
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 PTEIN	PTP FSM Timeout Error Injection Indicates the status of error injection for PTP FSM timeout. 0b - Disabled 1b - Enabled
11 ATEIN	APP FSM Timeout Error Injection Indicates the status of APP FSM timeout error injection. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
10 —	Reserved
9 RTEIN	Receive FSM Timeout Error Injection Indicates the status of receive FSM timeout error injection. 0b - Disabled 1b - Enabled
8 TTEIN	Transmit FSM Timeout Error Injection Indicates the status of transmit FSM timeout eror injection. 0b - Disabled 1b - Enabled
7-2 —	Reserved
1 PRTYEN	Parity Enable Indicates whether the FSM parity feature is enabled. 0b - Disabled 1b - Enabled
0 TMOUTEN	Time Out Enable Indicates the status of the FSM timeout feature. 0b - Disabled 1b - Enabled

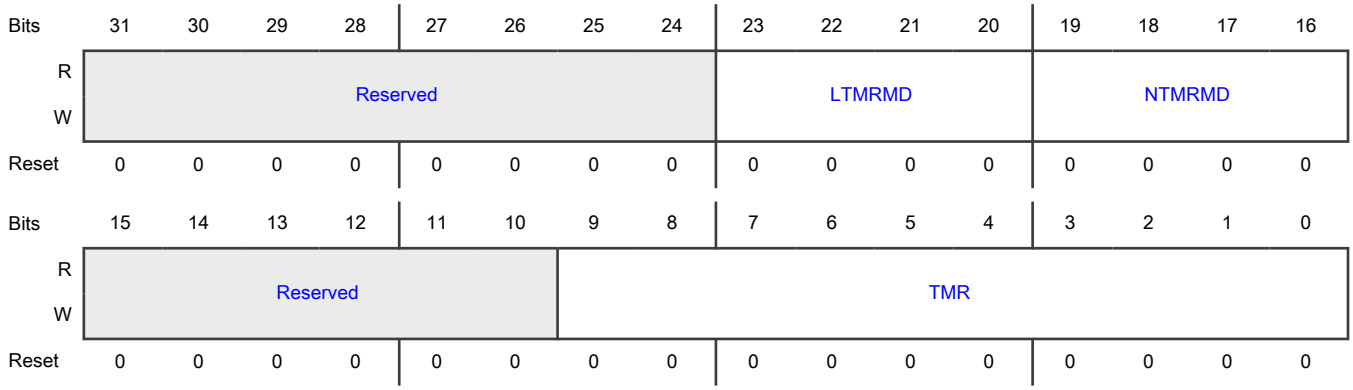
72.18.43 MAC FSM ACT Timer (MAC_FSM_ACT_Timer)

Offset

Register	Offset
MAC_FSM_ACT_Timer	14Ch

Function
Is used to select the FSM and interface timeout values.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 LTMRMD	Large Mode Timeout Value Provides the timeout duration value to be used for large mode FSM and other interface timeouts. 0000b - Timer disabled 0001b - 1 us 0010b - 1.024 ms (~4 ms) 0011b - 16.384 ms (~16 ms) 0100b - 65.536 ms (~64 ms) 0101b - 262.144 ms (~256 ms) 0110b - 1.048 sec (~1 sec) 0111b - 4.194 sec (~4sec) 1000b - 16.777 sec (~16 sec) 1001b - 33.554 sec (~32 sec) 1010b - 67.108 sec (~64 sec) 1011b - Reserved
19-16 NTMRMD	Normal Mode Timeout Value Provides the timeout duration value to be used for normal mode FSM and other interface timeouts. 0000b - Timer disabled 0001b - 1 us 0010b - 1.024 ms (~4 ms) 0011b - 16.384 ms (~16 ms)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - 65.536 ms (~64 ms) 0101b - 262.144 ms (~256 ms) 0110b - 1.048 sec (~1 sec) 0111b - 4.194 sec (~4 sec) 1000b - 16.777 sec (~16 sec) 1001b - 33.554 sec (~32 sec) 1010b - 67.108 sec (~64 sec) 1011b - Reserved
15-10 —	Reserved
9-0 TMR	CSR Clocks For 1 µs Tic Indicates the number of CSR clocks required to generate 1 µs tic.

72.18.44 SCS_REG 1 (SCS_REG1)

Offset

Register	Offset
SCS_REG1	150h

Function
Is an NXP-reserved register for internal use.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MAC_SCS1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MAC_SCS1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MAC_SCS1	MAC SCS 1 Is reserved for NXP internal use. The value of this field must always be 0 unless NXP instructs you otherwise. Writing 1 to any of the bits of this field might cause expected chip behavior.

72.18.45 MAC MDIO Address (MAC_MDIO_Address)

Offset

Register	Offset
MAC_MDIO_Address	200h

Function
Controls the management cycles to external PHY through a management interface.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	NTC			CR				Reserved			SKAP	GOC_1	GOC_0	C45E	GB
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 PSE	Preamble Suppression Enable Indicates the status of preamble suppression. • If this field is 1, SMA suppresses the 32-bit preamble and transmits MDIO frames with only 1 preamble bit. • If this field is 0, the MDIO frame always includes 32 bits of preamble as defined in the IEEE specification.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
26 BTB	Back-To-Back Transactions Indicates the status of back-to-back transactions. When BTB = 1 and NTC > 0, MAC indicates the completion of a read or write command at the end of a frame transfer (before the trailing clocks are transmitted). Therefore, you can initiate the next command, and that command executes immediately irrespective of the number of trailing clocks generated for the previous frame. When BTB = 0, the read or write command completes (GB = 0) only after the trailing clocks generate. In this mode, you must ensure that: • NTC is always generated after each frame. • BTB must not be equal to 1 when NTC = 0. 0b - Disabled 1b - Enabled
25-21 PA	Physical Layer Address Indicates whether MAC accesses clause 22 PHY devices (out of 32) or clause 45 capable PHYs (out of 32).
20-16 RDA	Register Or Device Address Indicates whether MAC accesses clause 22 PHY devices (out of 32) or clause 45 capable PHYs (out of 32).
15 —	Reserved
14-12 NTC	Number Of Trailing Clocks Controls the number of trailing clock cycles generated on the gmii_mdc_o (MDC) signal after the end of MDIO frame transmission. The valid values can range from 0 to 7. Writing 3h to this field indicates that there are additional three clock cycles on the MDC line after the end of MDIO frame transfer.
11-8 CR	CSR Clock Range Determines the frequency of the MDC clock according to the CSR clock frequencies used in your design: • 0000: CSR clock = 60-100 MHz; MDC clock = CSR clock/42 • 0001: CSR clock = 100-150 MHz; MDC clock = CSR clock/62 • 0010: CSR clock = 20-35 MHz; MDC clock = CSR clock/16 • 0011: CSR clock = 35-60 MHz; MDC clock = CSR clock/26 • 0100: CSR clock = 150-250 MHz; MDC clock = CSR clock/102 • 0101: CSR clock = 250-300 MHz; MDC clock = CSR clock/124

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0110: CSR clock = 300-500 MHz; MDC clock = CSR clock/204 • 0111: CSR clock = 500-800 MHz; MDC clock = CSR clock/324 When bit 3 of this field is 0, the suggested CSR clock-frequency range applicable to each value ensures that the MDC clock frequency is approximately between 1.0 MHz and 2.5 MHz. When bit 3 of this field is 1, you can achieve a higher MDC clock frequency than the IEEE 802.3 frequency limit of 2.5 MHz, and program a clock divider of a lower value. For example, if the CSR clock frequency is 100 MHz and you write 1010 to this field, the resultant MDC clock frequency equals 12.5 MHz, which is above the range specified in IEEE 802.3. Program these values only if the interfacing chips support faster MDC clocks: • 1000: CSR clock/4 • 1001: CSR clock/6 • 1010: CSR clock/8 • 1011: CSR clock/10 • 1100: CSR clock/12 • 1101: CSR clock/14 • 1110: CSR clock/16 • 1111: CSR clock/18 These bits are not used for accessing RevMII. They are read-only if the RevMII interface is selected as a single PHY interface.
7-5 —	Reserved
4 SKAP	Skip Address Packet Indicates the status of the skip address packet. If this field is 1, SMA does not send the address packets before read, write, or post-read increment address packets. This field is valid only when C45E = 1. 0b - Disabled 1b - Enabled
3 GOC_1	GMII Operation Command 1 Indicates the status of GMII operation command 1. GMII operation command 1 represents the higher bit of the operation command to PHY or RevMII. The fields GOC_1 and GOC_O indicate these values: • 00: Reserved • 01: Write • 10: Post-read increment address for clause 45 PHY • 11: Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When clause 22 PHY or RevMII is enabled, only write and read commands are valid. 0b - Disabled 1b - Enabled
2 GOC_0	GMII Operation Command 0 Indicates the status of GMII operation command 0. GMII operation command 0 represents the lower bit of the operation command to PHY or RevMII. When in SMA mode (MDIO master), this field, along with the GOC_1 field, determines the operation to be performed to PHY. This field is read-only and tied to 1 when only RevMII is selected in configuration. 0b - Disabled 1b - Enabled
1 C45E	Clause 45 PHY Enable Indicates the status of clause 45 PHY. • If this field is 1, clause 45 capable PHY is connected to MDIO. • If this field is 0, clause 22 capable PHY is connected to MDIO. 0b - Disabled 1b - Enabled
0 GB	GMII Busy Indicates the status of GMII busy. Writing 1 to this field instructs SMA to initiate a read or write access to the MDIO slave. MAC writes 0 to the field after the MDIO frame transfer is complete. Therefore, you must not write or change any of the fields in the MAC_MDIO_Address and MAC_MDIO_Data registers as long as the value of this field is 1. For write transfers, before writing 1 to this field, you must write a 16-bit data to MAC_MDIO_Data[GD] and MAC_MDIO_Data[RA] if C45E = 1. If C45E = 1, you must also write to MAC_MDIO_Data[RA] before initiating a read transfer. After a read transfer completes (GB = 0), a data read from the PHY register is valid in MAC_MDIO_Data[GD]. **NOTE** Even if the addressed PHY is not present, there is no change in the functionality of this field. Access restrictions apply to this field. It clears automatically and writing 0 to it has no effect. 0b - Disabled 1b - Enabled

72.18.46 MAC MDIO Data (MAC_MDIO_Data)

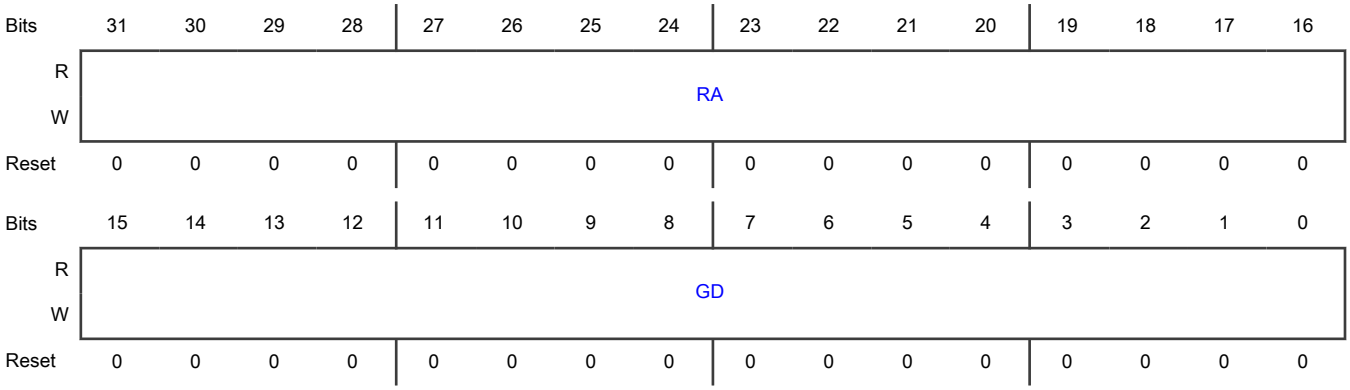
Offset

Register	Offset
MAC_MDIO_Data	204h

Function

Is used to store the write data to be written to the PHY register located at the address specified in MAC MDIO Address (MAC_MDIO_Address). This register also stores the read data from the PHY register located at the address that MAC MDIO Address (MAC_MDIO_Address) specifies.

Diagram



Fields

Field	Function
31-16 RA	Register Address Contains the PHY register address intended for the MDIO frame. This field is valid only when MAC_MDIO_Address[C45E] = 1.
15-0 GD	GMII Data Contains the 16-bit data value read from PHY or RevMII after a management read operation or the 16-bit data value to be written to PHY or RevMII before a management write operation.

72.18.47 MAC CSR Software Control (MAC_CSR_SW_Ctrl)

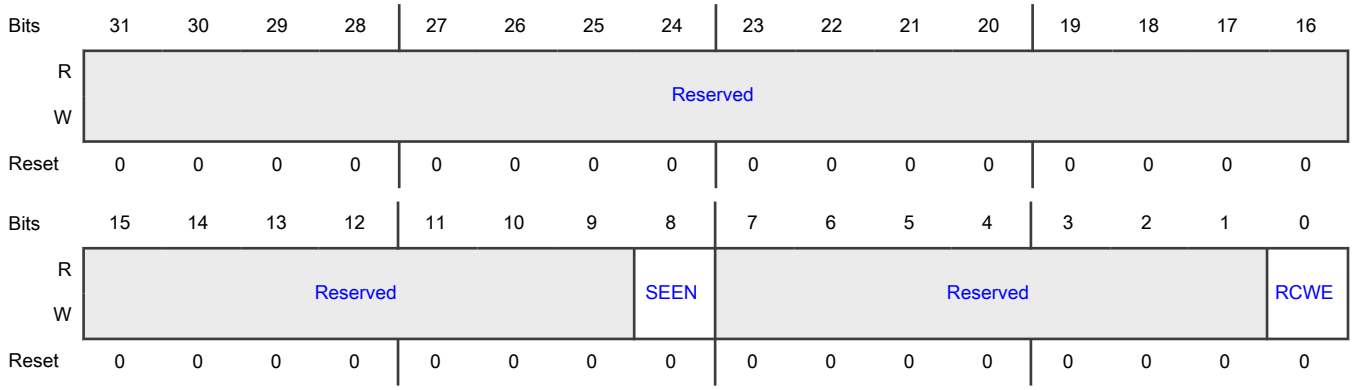
Offset

Register	Offset
MAC_CSR_SW_Ctrl	230h

Function

Contains software programmable controls for changing the CSR access response and clearing the status bits on read.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 SEEN	Slave Error Response Enable Indicates the status of the slave error response. • If this field is 1, MAC responds with a slave error for accesses to reserved registers in the CSR space. • If this field is 0, MAC sends an "okay" response to registers accessed from the CSR space. 0b - Disabled 1b - Enabled
7-1 —	Reserved
0 RCWE	Enable Register Write 1 To Clear (W1C) Enables and disables the W1C feature for some registers. When this field is 1, the access mode of some of the other fields changes from R2C to W1C. When this field is 0, the access mode of those fields remains R2C. 0b - Disabled 1b - Enabled

72.18.48 MAC FPE Control STS (MAC_FPE_CTRL_STS)

Offset

Register	Offset
MAC_FPE_CTRL_STS	234h

Function

Controls the operation of frame preemption.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												S1_SET_0	SRSP	SVER	EFPE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 TRSP	Transmitted Respond Frame Indicates whether the respond frame is transmitted. This field becomes 1 when a respond frame is transmitted (triggered by writing 1 to the SRSP field). An interrupt can be generated for this event if you write 1 to MAC_Interrupt_Enable[FPEIE]. Access restrictions apply to this field. TRSP becomes a W1C field when MAC_CSR_SW_Ctrl[RCWE] = 1, and becomes an R2C field when MAC_CSR_SW_Ctrl[RCWE] = 0. Thus, TRSP becomes 0 sometimes when you read it and sometimes when you write 1 to it. It automatically becomes 1 on an internal event. 0b - Not transmitted 1b - Transmitted
18 TVER	Transmitted Verify Frame Indicates whether the verify frame is transmitted. This field becomes 1 when a verify frame is transmitted (triggered by writing 1 to the SVER field). An interrupt can be generated for this event if you write 1 to MAC_Interrupt_Enable[FPEIE]. Access restrictions apply to this field. The field clears on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. It automatically becomes 1 on an internal event. 0b - Not transmitted 1b - Transmitted
17 RRSP	Received Respond Frame Indicates whether the respond frame is received.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 1 when a respond frame is received. An interrupt can be generated for this event if you write 1 to MAC_Interrupt_Enable[FPEIE]. Access restrictions apply to this field. The field clears on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. It automatically becomes 1 on an internal event. 0b - Not received 1b - Received
16 RVER	Received Verify Frame Indicates whether the verify frame is received. This field becomes 1 when a verify frame is received. An interrupt can be generated on the transmission of a packet if you write 1 to MAC_Interrupt_Enable[FPEIE]. Access restrictions apply to this field. The field clears on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. It automatically becomes 1 on an internal event. 0b - Not received 1b - Received
15-4 —	Reserved
3 S1_SET_0	S1 SET 0 Is reserved for NXP internal use and must always be 0 unless instructed by NXP. Writing 1 to this field may cause unexpected behavior.
2 SRSP	Send Respond mPacket Indicates the status of the send respond mPacket. If this field is 1, it indicates the hardware to send a respond mPacket. The hardware resets the field after sending the respond mPacket. Access restrictions apply to this field. It clears automatically and writing 0 to it has no effect. 0b - Disabled 1b - Enabled
1 SVER	Send Verify mPacket Enables and disables the sending of a verify mPacket. Write 1 to this field to command EMAC to send a verify mPacket. EMAC writes 0 to this field after sending the verify mPacket. Access restrictions apply to this field. It clears automatically and writing 0 to it has no effect. 0b - Disabled 1b - Enabled
0	Enable Transmit Frame Preemption

Table continues on the next page...

Table continued from the previous page...

Field	Function
EFPE	Enables or disables the frame preemption transmit functionality. 0b - Disabled 1b - Enabled

72.18.49 MAC Presentation Time (MAC_Presn_Time_ns)

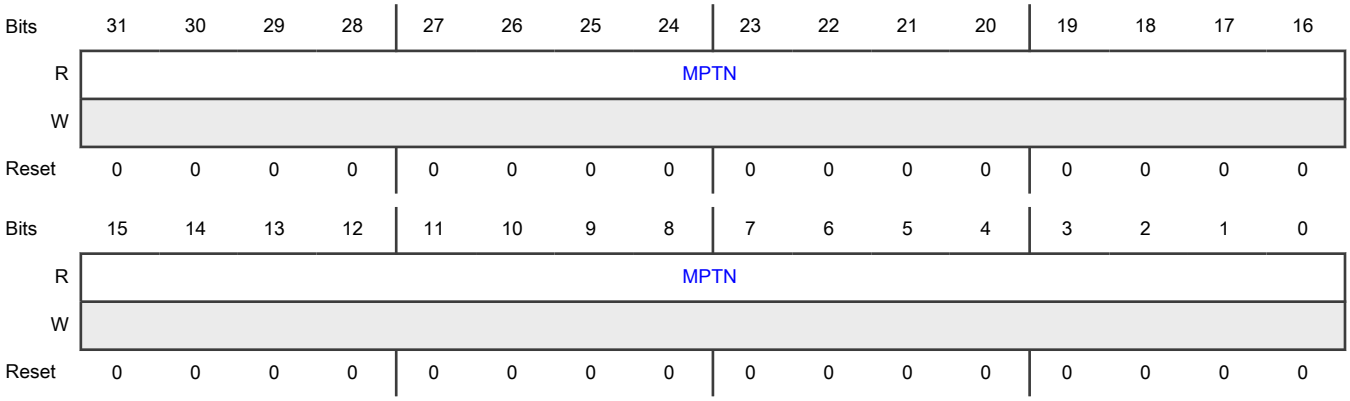
Offset

Register	Offset
MAC_Presn_Time_ns	240h

Function

Contains the 32-bit binary rollover equivalent time of the PTP system time (in nanoseconds) and exists when DWC_EQOS_FLEXI_PPS_OUT_EN is configured.

Diagram



Fields

Field	Function
31-0 MPTN	MAC 1722 Presentation Time (In Nanoseconds) Indicates the value of the 32-bit binary rollover equivalent time of the PTP System Time (in ns).

72.18.50 MAC Presentation Time Update (MAC_Presn_Time_Updt)

Offset

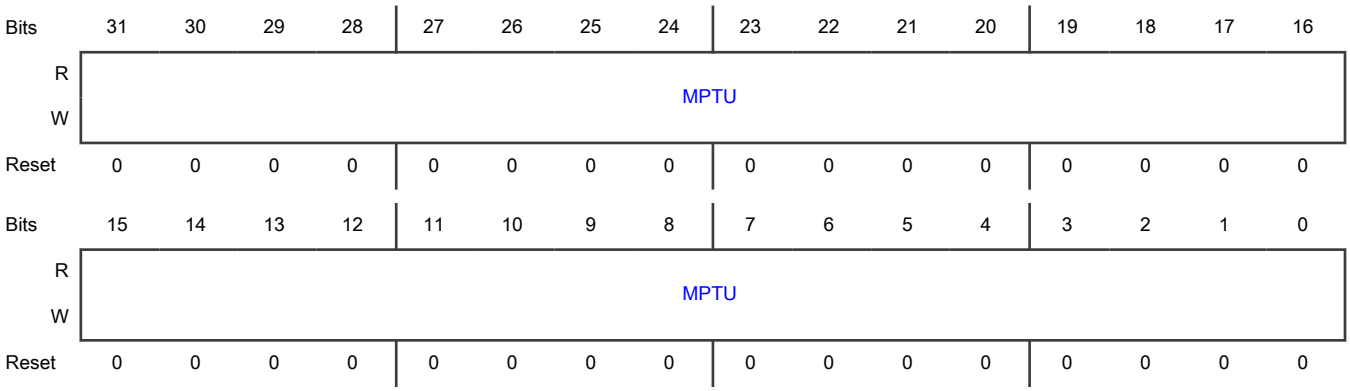
Register	Offset
MAC_Presn_Time_Updt	244h

Function

Holds the 32-bit value of MAC 1722 presentation time (in ns), which must be added to the current presentation time counter value. Initialization happens when MAC_Timestamp_Control[TSINIT] = 1, and an update happens when MAC_Timestamp_Control[TSUPDT] = 1.

Exists when DWC_EQOS_FLEXI_PPS_OUT_EN is configured.

Diagram



Fields

Field	Function
31-0 MPTU	MAC 1722 Presentation Time Update Holds the initial value or the update value for the presentation time. When used for an update, this field holds the 32-bit value (in ns), which must be added to the current presentation time counter value. Initialization happens when MAC_Timestamp_Control[TSINIT] = 1, and an update happens when MAC_Timestamp_Control[TSUPDT] = 1. When MAC_System_Time_Nanoseconds_Update[ADDSUB] = 1, this value is directly used for subtraction.

72.18.51 MAC Address 0 High (MAC_Address0_High)

Offset

Register	Offset
MAC_Address0_High	300h

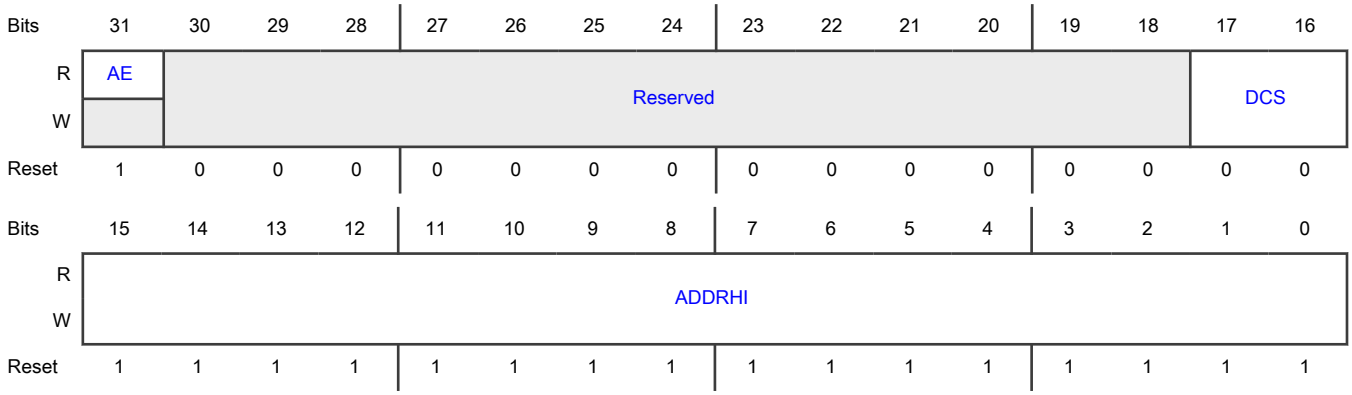
Function

Holds the upper 16 bits of the first 6-byte MAC address of the station.

The first DA byte that is received on the (G)MII interface corresponds to the LS byte (bits [7:0]) of MAC Address 0 Low (MAC_Address0_Low). For example, if 112233445566h is received (11h in lane 0 of the first column) on (G)MII as the destination address, MAC Address 0 Low (MAC_Address0_Low)[47:0] is compared to 665544332211h.

If MAC address registers are configured to be double-synchronized with the (G)MII clock domains, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of MAC Address 0 Low (MAC_Address0_Low) are written to. For proper synchronization updates, the consecutive writes to MAC Address 0 Low (MAC_Address0_Low) must be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31 AE	Address Enable Enables MAC address 0b - Disabled and invalid (the field's value must always be 1) 1b - Enabled
30-18 —	Reserved
17-16 DCS	DMA Channel Select If MAC_Ext_Configuration[PDC] = 0, this field contains the binary representation of the DMA channel number to which a receive packet whose DA matches MAC Address 0 Low (MAC_Address0_Low) content is routed. If MAC_Ext_Configuration[PDC] = 1, this field contains the one-hot representation of one or more DMA channel numbers to which a receive packet whose DA matches MAC Address 0 Low (MAC_Address0_Low) content is routed.
15-0 ADDRHI	MAC Address 0 [47:32] Contains the upper 16 bits [47:32] of the first 6-byte MAC address. MAC uses this field for filtering the received packets and inserting the MAC address in the transmit flow control (pause) packets.

72.18.52 MAC Address 0 Low (MAC_Address0_Low)

Offset

Register	Offset
MAC_Address0_Low	304h

Function

Holds the lower 32 bits of the 6-byte first MAC address of the station.

Diagram

Fields

Field	Function
31-0 ADDRLO	MAC Address 0 [31:0] Contains the lower 32 bits of the first 6-byte MAC address. MAC uses this field to filter the received packets and insert the MAC address in the transmit flow control (pause) packets.

72.18.53 MAC Address 1 High (MAC_Address1_High)

Offset

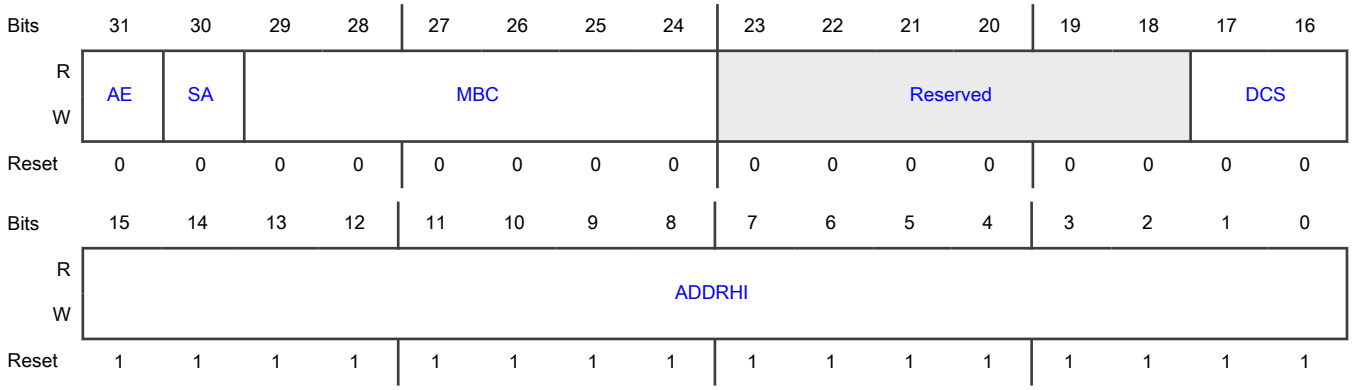
Register	Offset
MAC_Address1_High	308h

Function

Holds the upper 16 bits of the second 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized with the (G)MII clock domains, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of MAC Address 1 Low (MAC_Address1_Low) are written to. For proper synchronization updates, the consecutive writes to MAC Address 1 Low (MAC_Address1_Low) must be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31 AE	Address Enable Indicates whether the MAC address is enabled or ignored. If this field is 1, the address filter module uses the second MAC address for perfect filtering. If this field is 0, the address filter module ignores the address used for filtering. 0b - Ignored 1b - Enabled
30 SA	Source Address Indicates whether the MAC address is compared to the source or destination address. If this field is 1, MAC Address 1[47:0] is compared to the SA fields of the received packet. If this field is 0, MAC Address 1[47:0] is compared to the DA fields of the received packet. 0b - Destination address 1b - Source address
29-24 MBC	Mask Byte Control Contains mask control bits for comparing each of the MAC address bytes. If this field is 1, MAC does not compare the corresponding byte of the received DA or SA with the contents of the MAC address 1 registers. Each bit controls the masking of the bytes as follows: • Bit 29: MAC Address 1 High[15:8] • Bit 28: MAC Address 1 High[7:0] • Bit 27: MAC Address 1 Low[31:24] • - .. • Bit 24: MAC Address 1 Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-18	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
17-16 DCS	DMA Channel Select If MAC_Ext_Configuration[PDC] = 0, this field contains the binary representation of the DMA channel number to which a receive packet whose DA matches MAC Address 1 Low (MAC_Address1_Low) content is routed. If MAC_Ext_Configuration[PDC] = 1, this field contains the one-hot representation of one or more DMA Channel numbers to which a receive packet whose DA matches MAC Address 1 Low (MAC_Address1_Low) content is routed.
15-0 ADDRHI	MAC Address 1 [47:32] Contains the upper 16 bits [47:32] of the second 6-byte MAC address.

72.18.54 MAC Address 1 Low (MAC_Address1_Low)

Offset

Register	Offset
MAC_Address1_Low	30Ch

Function

Holds the lower 32 bits of the second 6-byte MAC address of the station.

Diagram

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

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

Fields

Field	Function
31-0	MAC Address 1 [31:0] Contains the lower 32 bits of the second 6-byte MAC address.

Table continues on the next page...

Field	Function
ADDRLO	The contents of this field are undefined until you load them after the initialization process.

72.18.55 MAC Address 2 High (MAC_Address2_High)

Offset

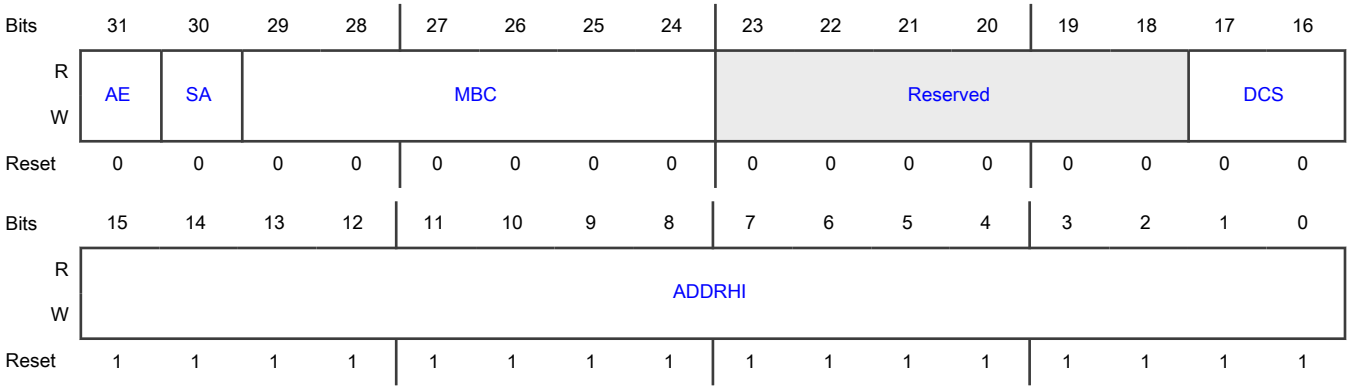
Register	Offset
MAC_Address2_High	310h

Function

Holds the upper 16 bits of the second 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized with the (G)MII clock domains, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of MAC Address 1 Low (MAC_Address1_Low) are written to. For proper synchronization updates, the consecutive writes to MAC Address 1 Low (MAC_Address1_Low) must be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31 AE	Address Enable Indicates whether the MAC address is enabled or ignored. If this field is 1, the address filter module uses the second MAC address for perfect filtering. If this field is 0, the address filter module ignores the address for filtering. 0b - Ignored 1b - Enabled
30 SA	Source Address Indicates whether the MAC address is compared to the source or destination address.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 1, MAC Address 1[47:0] is compared to the SA fields of the received packet. If this field is 0, MAC Address 1[47:0] is compared to the DA fields of the received packet. 0b - Destination address 1b - Source address
29-24 MBC	Mask Byte Control Contains mask control bits for comparing each of the MAC address bytes. If this field is 1, MAC does not compare the corresponding byte of the received DA or SA with the contents of MAC address 1 registers. Each bit controls the masking of the bytes as follows: • Bit 29: MAC Address 1 High[15:8] • Bit 28: MAC Address 1 High[7:0] • Bit 27: MAC Address 1 Low[31:24] • - .. • Bit 24: MAC Address 1 Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-18 —	Reserved
17-16 DCS	DMA Channel Select If MAC_Ext_Configuration[PDC] = 0, this field contains the binary representation of the DMA channel number to which a receive packet whose DA matches MAC Address 2 Low (MAC_Address2_Low) content is routed. If MAC_Ext_Configuration[PDC] = 1, this field contains the one-hot representation of one or more DMA Channel numbers to which a receive packet whose DA matches MAC Address 2 Low (MAC_Address2_Low) content is routed.
15-0 ADDRHI	MAC Address 1 [47:32] Contains the upper 16 bits [47:32] of the second 6-byte MAC address.

72.18.56 MAC Address 2 Low (MAC_Address2_Low)

Offset

Register	Offset
MAC_Address2_Low	314h

Function

Holds the lower 32 bits of the second 6-byte MAC address of the station.

Diagram

Fields

Field	Function
31-0 ADDRLO	MAC Address 1 [31:0] Contains the lower 32 bits of the second 6-byte MAC address. The content of this field remains undefined until you load it after the initialization process.

72.18.57 MMC Control (MMC_Control)

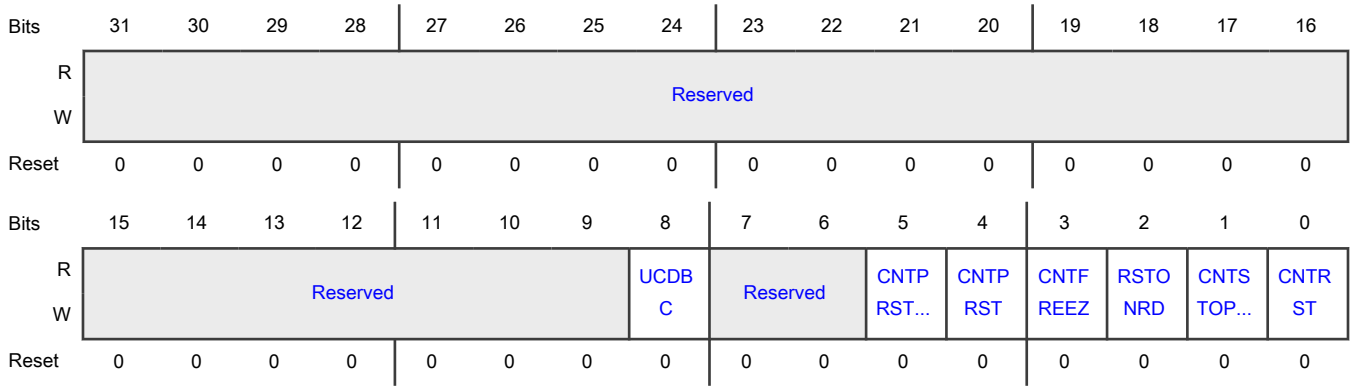
Offset

Register	Offset
MMC_Control	700h

Function

Establishes the operating mode for MMC.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 UCDBC	Update MMC Counters For Dropped Broadcast Packets Indicates the status of MMC counters for dropped broadcast packets. • If this field is 1, MAC updates all the related MMC counters for broadcast packets that are dropped because MAC_Packet_Filter[DBF] = 1. • If this field is 0, the MMC counters are not updated for dropped broadcast packets. **NOTE** The CNTRST field has a higher priority than the CNTPRST field. Therefore, if you write 1 to both these fields in the same write cycle, all the counters are cleared and the CNTPRST field becomes 0. 0b - Disabled 1b - Enabled
7-6 —	Reserved
5 CNTPRSTLVL	Full-Half Preset Indicates whether full-half preset is enabled or disabled. • If this field is 0 and the CNTPRST field is 1, all the MMC counters are preset to the almost-half value. All octet counters are preset to 7FFF_F800h (half 2 Kbytes) and all packet counters are preset to 7FFF_FFF0h (half 16). • If this field is 1 and the CNTPRST field is 1 too, all MMC counters are preset to the almost-full value. All octet counters are preset to FFFF_F800h (full 2 Kbytes) and all packet counters are preset to FFFF_FFF0h (full 16). For 16-bit counters, the almost-half preset values are 7800h and 7FF0h for the respective octet and packet counters. Similarly, the almost-full preset values for the 16-bit counters are 0xF800 and 0xFFF0. 0b - Disabled 1b - Enabled
4 CNTPRST	Counters Preset Indicates whether MMC counter preset is enabled or disabled. If this field is 1, all the counters are initialized or preset to almost full or almost half according to the settings of the CNTPRSTLVL field. This field clears automatically after 1 clock cycle, and along with the CNTPRSTLVL field, is useful for debugging and testing the assertion of interrupts because of the MMC counter becoming half-full or full. Access restrictions apply to this field, which clears automatically. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
3 CNTFREEZ	MMC Counter Freeze Indicates the status of MMC counter freeze. If this field is 1, the field freezes all the MMC counters to their current values. Until this field becomes 0, no MMC counter is updated because of any transmitted or received packets. If an MMC counter is read when RSTONRD = 1, that counter is also cleared in this mode. 0b - Disabled 1b - Enabled
2 RSTONRD	Reset On Read Indicates whether the reset on read for MMC counters is enabled or disabled. If this field is 1, MMC counters are reset to 0 after read (they clear automatically after reset). The counters are cleared when the least significant byte lane (bits [7:0]) is read. 0b - Disabled 1b - Enabled
1 CNTSTOPRO	Counter Stop Rollover Indicates the status of counter stop rollover. If this field is 1, the counter does not roll over to 0 after reaching the maximum value. 0b - Disabled 1b - Enabled
0 CNTRST	Counters Reset Indicates whether MMC counters are reset. If this field is 1, all the counters are reset. This field clears automatically after 1 clock cycle. Access restrictions apply to this field that clears automatically. 0b - Counters are not reset 1b - All counters are reset

72.18.58 MMC Receive Interrupt (MMC_Rx_Interrupt)

Offset

Register	Offset
MMC_Rx_Interrupt	704h

Function

Maintains the interrupts generated from all receive statistics counters when the following conditions occur:

- Receive statistic counters reach half of their maximum values (8000_0000h for a 32-bit counter and 8000h for a 16-bit counter).
- Receive statistic counters cross their maximum values (FFFF_FFFFh for a 32-bit counter and FFFFh for a 16-bit counter).

If MMC_Control[CNTSTOPRO] = 1, interrupts are set but the counter remains at all-ones. This is a 32-bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (bits [7:0]) of the respective counter must be read to clear the interrupt bit.

NOTE

R_SS_RC means that this register bit is set internally, and is cleared when the counter register is read.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				Reserved	Reserved	RXCTRLP...	RXRCVER...	RXWDOGP...	RXVLANG...	RXFOVPIS	RXPAUSP...	RXORANG...	RXLENER...	RXUCGPIS	RX1024T...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RX512T1...	RX256T5...	RX128T2...	RX65T12...	RX64OCT...	RXOSIZE...	RXUSIZE...	RXJABER...	RXRUNTP...	RXALGNE...	RXCRCER...	RXMCGPIS	RXBCGPIS	RXGOCTIS	RXGBOCT...	RXGBPKT...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 RXCTRLPIS	MMC Receive Control Packet Counter Interrupt Status Indicates whether the status of the MMC receive control packet counter interrupt is detected. This field becomes 1 when the rxctrlpackets_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 RXRCVERRPIS	MMC Receive Error Packet Counter Interrupt Status Indicates whether the status of the MMC receive error packet counter interrupt is detected. This field becomes 1 when the rxrcverror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
23 RXWDOGPIS	MMC Receive Watchdog Error Packet Counter Interrupt Status Indicates whether the status of the MMC receive watchdog error packet counter interrupt is detected. This field becomes 1 when the rxwatchdog error counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
22 RXVLANGBPIS	MMC Receive VLAN Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive VLAN good bad packet counter interrupt is detected. This field becomes 1 when the rxvlanpackets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
21 RXFOVPIS	MMC Receive FIFO Overflow Packet Counter Interrupt Status Indicates whether the status of the MMC receive FIFO overflow packet counter interrupt is detected. This field becomes 1 when the rxfifooverflow counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
20 RXPAUSPIS	MMC Receive Pause Packet Counter Interrupt Status Indicates whether the status of the MMC receive pause packet counter interrupt is detected. This field becomes 1 when the rxpausepackets counter reaches half of its maximum value or the maximum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
19 RXORANGEPIS	MMC Receive Out-Of-Range Error Packet Counter Interrupt Status Indicates whether the status of the MMC receive out-of-range error packet counter interrupt is detected. This field becomes 1 when the rxoutofrangetype counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
18 RXLENERPIS	MMC Receive Length Error Packet Counter Interrupt Status Indicates whether the status of the MMC receive length error packet counter interrupt is detected. This field becomes 1 when the rxlengtherror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
17 RXUCGPIS	MMC Receive Unicast Good Packet Counter Interrupt Status Indicates whether the status of the MMC receive unicast good packet counter interrupt is detected. This field becomes 1 when the rxunicastpackets_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
16 RX1024TMAXOCTGBPIS	MMC Receive 1024 To Maximum Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive 1024 to maximum octet good bad packet counter interrupt is detected. This field becomes 1 when the rx1024tomaxoctets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected
15 RX512T1023OCTGBPIS	MMC Receive 512 To 1023 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive 512 to 1023 octet good bad packet counter interrupt is detected. This field becomes 1 when the rx512to1023octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
14 RX256T511OCTGBPIS	MMC Receive 256 To 511 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive 256 to 511 octet good bad packet counter interrupt is detected. This field becomes 1 when the rx256to511octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
13 RX128T255OCTGBPIS	MMC Receive 128 To 255 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive 128 to 255 octet good bad packet counter interrupt is detected. This field becomes 1 when the rx128to255octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
12 RX65T127OCTGBPIS	MMC Receive 65 To 127 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive 65 to 127 octet good bad packet counter interrupt is detected. The field becomes 1 when the rx65to127octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected
11 RX64OCTGBPIS	MMC Receive 64 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive 64-octet good bad packet counter interrupt is detected. The field becomes 1 when the rx64octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
10 RXOSIZEGPIS	MMC Receive Oversize Good Packet Counter Interrupt Status Indicates whether the status of the MMC receive oversize good packet counter interrupt is detected. This field becomes 1 when the rxoversize_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
9 RXUSIZEGPIS	MMC Receive Undersize Good Packet Counter Interrupt Status Indicates whether the status of MMC receive undersize good packet counter interrupt is detected. This field becomes 1 when the rxundersize_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
8 RXJABERPIS	MMC Receive Jabber Error Packet Counter Interrupt Status Indicates whether the status of MMC receive jabber error packet counter interrupt is detected. This field becomes 1 when the rxjabbererror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 RXRUNTPIS	MMC Receive Runt Packet Counter Interrupt Status Indicates whether the status of the MMC receive runt packet counter interrupt is detected. This field becomes 1 when the rxrunterror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
6 RXALGNERPIS	MMC Receive Alignment Error Packet Counter Interrupt Status Indicates whether the status of the MMC receive alignment error packet counter interrupt is detected. This field becomes 1 when the rxalignmenterror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
5 RXCRCERPIS	MMC Receive CRC Error Packet Counter Interrupt Status Indicates whether the status of the MMC receive CRC error packet counter interrupt is detected. This field becomes 1 when the rxcrcerror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
4 RXMCGPIS	MMC Receive Multicast Good Packet Counter Interrupt Status Indicates whether the status of the MMC receive multicast good packet counter interrupt is detected. This field becomes 1 when the rxmulticastpackets_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
3 RXBCGPIS	MMC Receive Broadcast Good Packet Counter Interrupt Status Indicates whether the status of the MMC receive broadcast good packet counter interrupt is detected. This field becomes 1 when the rxbroadcastpackets_g counter reaches half of its maximum value or the maximum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
2 RXGOCTIS	MMC Receive Good Octet Counter Interrupt Status Indicates whether the status of the MMC receive good octet counter interrupt is detected. This field becomes 1 when the rxoctetcount_g counter reaches half of its maximum value or its maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
1 RXGBOCTIS	MMC Receive Good Bad Octet Counter Interrupt Status Indicates whether the status of the MMC receive good bad octet counter interrupt is detected. This field becomes 1 when the rxoctetcount_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
0 RXGBPKTIS	MMC Receive Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive good bad packet counter interrupt is detected. This field becomes 1 when the rxpacketcount_gb counter reaches half of its maximum value or its maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

72.18.59 MMC Transmit Interrupt (MMC_Tx_Interrupt)

Offset

Register	Offset
MMC_Tx_Interrupt	708h

Function

Maintains the interrupts generated from all the transmit statistics counters.

This register maintains those interrupts that are generated when transmit statistic counters:

- Reach half of their maximum values (8000_0000h for a 32-bit counter and 0x8000 for a 16-bit counter)
- Cross their maximum values (FFFF_FFFFh for a 32-bit counter and FFFFh for a 16-bit counter)

If MMC_Control[CNTSTOPRO] = 1, the interrupts are set but the counter remains at all-ones.

This is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read.

The least significant byte lane (bits [7:0]) of the respective counter must be read to clear the interrupt bit.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				Reserved	Reserved	TXOSIZE...	TXVLANG...	TXPAUSP...	TXEXDEF...	TXGPKTIS	TXGOCTIS	TXCARER...	TXEXCOL...	TXLATCO...	TXDEFPIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXMCOLG...	TXSCOLG...	TXUFLOW...	TXBCGBP...	TXMCGBP...	TXUCGBP...	TX1024T...	TX512T1...	TX256T5...	TX128T2...	TX65T12...	TX64OCT...	TXMCGPIS	TXBCGPIS	TXGBPKT...	TXGBOCT...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 TXOSIZEGPIS	MMC Transmit Oversize Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit oversize good packet counter interrupt is detected. This field becomes 1 when the txoversize_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Detected
24 TXVLANGPIS	MMC Transmit VLAN Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit VLAN good packet counter interrupt is detected. This field becomes 1 when the txvlanpackets_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
23 TXPAUSPIS	MMC Transmit Pause Packet Counter Interrupt Status Indicates whether the status of the MMC transmit pause packet counter interrupt is detected. This field becomes 1 when the txpausepacketserror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
22 TXEXDEFPIS	MMC Transmit Excessive Deferral Packet Counter Interrupt Status Indicates whether the status of the MMC transmit excessive deferral packet counter interrupt is detected. This field becomes 1 when the txexcessdef counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
21 TXGPKTIS	MMC Transmit Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit good packet counter interrupt is detected. This field becomes 1 when the txpacketcount_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
20	MMC Transmit Good Octet Counter Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
TXGOCTIS	Indicates whether the status of the MMC transmit good octet counter interrupt is detected. This field becomes 1 when the txoctetcount_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
19 TXCARERPIS	MMC Transmit Carrier Error Packet Counter Interrupt Status Indicates whether the status of the MMC transmit carrier error packet counter interrupt is detected. This field becomes 1 when the txcarriererror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
18 TXEXCOLPIS	MMC Transmit Excessive Collision Packet Counter Interrupt Status Indicates whether the status of the MMC transmit excessive collision packet counter interrupt is detected. This field becomes 1 when the txexesscol counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
17 TXLATCOLPIS	MMC Transmit Late Collision Packet Counter Interrupt Status Indicates whether the status of the MMC transmit late collision packet counter interrupt is detected. This field becomes 1 when the txlatecol counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
16 TXDEFPIS	MMC Transmit Deferred Packet Counter Interrupt Status Indicates whether the status of the MMC transmit deferred packet counter interrupt is detected. This field becomes 1 when the txdeferred counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected
15 TXMCOLGPIS	MMC Transmit Multiple Collision Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit multiple collision good packet counter interrupt is detected. This field becomes 1 when the txmulticol_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
14 TXSCOLGPIS	MMC Transmit Single Collision Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit single collision good packet counter interrupt is detected. This field becomes 1 when the txsinglecol_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
13 TXUFLOWERPIS	MMC Transmit Underflow Error Packet Counter Interrupt Status Indicates whether the status of the MMC transmit underflow error packet counter interrupt is detected. This field becomes 1 when the txunderflowerror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
12 TXBCGBPIS	MMC Transmit Broadcast Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit broadcast good bad packet counter interrupt is detected. This field becomes 1 when the txbroadcastpackets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 TXMCGBPIS	MMC Transmit Multicast Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit multicast good bad packet counter interrupt is detected. This field becomes 1 when the txmulticastpackets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
10 TXUCGBPIS	MMC Transmit Unicast Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit unicast good bad packet counter interrupt is detected. This field becomes 1 when the txunicastpackets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
9 TX1024TMAXOCTGBPIS	MMC Transmit 1024 To Maximum Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit 1024 to maximum octet good bad packet counter interrupt is detected. This field becomes 1 when the tx1024tomaxoctets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
8 TX512T1023OCTGBPIS	MMC Transmit 512 To 1023 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit 512 to 1023 octet good bad packet counter interrupt is detected. This field becomes 1 when the tx512to1023octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
7	MMC Transmit 256 To 511 Octet Good Bad Packet Counter Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
TX256T511OCTGBPIS	Indicates whether the status of the MMC transmit 256 to 511 octet good bad packet counter interrupt is detected. This field becomes 1 when the tx256to511octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
6 TX128T255OCTGBPIS	MMC Transmit 128 To 255 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit 128 to 255 octet good bad packet counter interrupt is detected. This field becomes 1 when the tx128to255octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
5 TX65T127OCTGBPIS	MMC Transmit 65 To 127 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit 65 to 127 octet good bad packet counter interrupt is detected. This field becomes 1 when the tx65to127octets_gb counter reaches half of its maximum value, and also when it reaches the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
4 TX64OCTGBPIS	MMC Transmit 64-Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit 64-octet good bad packet counter interrupt is detected. This field becomes 1 when the tx64octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
3 TXMCGPIS	MMC Transmit Multicast Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit multicast good packet counter interrupt is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 1 when the txmulticastpackets_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
2 TXBCGPIS	MMC Transmit Broadcast Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit broadcast good packet counter interrupt is detected. This field becomes 1 when the txbroadcastpackets_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
1 TXGBPKTIS	MMC Transmit Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit good bad packet counter interrupt is detected. This field becomes 1 when the txpacketcount_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
0 TXGBOCTIS	MMC Transmit Good Bad Octet Counter Interrupt Status Indicates whether the status of the MMC transmit good bad octet counter interrupt is detected. This field becomes 1 when the txoctetcount_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

72.18.60 MMC Receive Interrupt Mask (MMC_Rx_Interrupt_Mask)

Offset

Register	Offset
MMC_Rx_Interrupt_Mask	70Ch

Function

Maintains the masks for interrupts generated from all the receive statistic counters. These interrupts are generated when the receive statistic counters reach half of their maximum values or the maximum values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				Reserved	Reserved	RXCTRLP...	RXRCVER...	RXWDOGP...	RXVLANG...	RXFOVPIM	RXPAUSP...	RXORANG...	RXLENER...	RXUCGPIM	RX1024T...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RX512T1...	RX256T5...	RX128T2...	RX65T12...	RX64OCT...	RXOSIZE...	RXUSIZE...	RXJABER...	RXRUNTP...	RXALGNE...	RXCRCER...	RXMCGPIM	RXBCGPIM	RXGOCTIM	RXGBOCT...	RXGBPKT...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 RXCTRLPIM	MMC Receive Control Packet Counter Interrupt Mask Indicates the status of the MMC receive control packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxctrlpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
24 RXRCVERRPIM	MMC Receive Error Packet Counter Interrupt Mask Indicates the status of the MMC receive error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxrcverror counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
23	MMC Receive Watchdog Error Packet Counter Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXWDOGPIM	Indicates the status of the MMC receive watchdog error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxwatchdog counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
22 RXVLANGBPIM	MMC Receive VLAN Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive VLAN good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxvlanpackets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
21 RXFOVPIM	MMC Receive FIFO Overflow Packet Counter Interrupt Mask Indicates the status of the MMC receive FIFO overflow packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxfifooverflow counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
20 RXPAUSPIM	MMC Receive Pause Packet Counter Interrupt Mask Indicates the status of the MMC receive pause packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxpausepackets counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
19 RXORANGEPIM	MMC Receive Out-Of-Range Error Packet Counter Interrupt Mask Indicates the status of the MMC receive out-of-range error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxoutofrangetype counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
18 RXLENERPIM	MMC Receive Length Error Packet Counter Interrupt Mask Indicates the status of the MMC receive length error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxlengtherror counter reaches half of its maximum value or the maximum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
17 RXUCGPIM	MMC Receive Unicast Good Packet Counter Interrupt Mask Indicates the status of the MMC receive unicast good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxunicastpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
16 RX1024TMAXOCTGBPIM	MMC Receive 1024 To Maximum Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive 1024 to maximum octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rx1024tomaxoctets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
15 RX512T1023OCTGBPIM	MMC Receive 512 To 1023 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive 512 to 1023 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rx512to1023octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
14 RX256T511OCTGBPIM	MMC Receive 256 To 511 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive 256 to 511 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rx256to511octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
13 RX128T255OCTGBPIM	MMC Receive 128 To 255 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive 128 to 255 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rx128to255octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
12	MMC Receive 65 To 127 Octet Good Bad Packet Counter Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
RX65T127OCTGBPIM	Indicates the status of the MMC receive 65 to 127 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rx65to127octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
11 RX64OCTGBPIM	MMC Receive 64-Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive 64-octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rx64octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
10 RXOSIZEGPIM	MMC Receive Oversize Good Packet Counter Interrupt Mask Indicates the status of the MMC receive oversize good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxoversize_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
9 RXUSIZEGPIM	MMC Receive Undersize Good Packet Counter Interrupt Mask Indicates the status of the MMC receive undersize good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxundersize_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
8 RXJABERPIM	MMC Receive Jabber Error Packet Counter Interrupt Mask Indicates the status of the MMC receive jabber error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxjabbererror counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
7 RXRUNTPIM	MMC Receive Runt Packet Counter Interrupt Mask Indicates the status of the MMC receive runt packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxrunterror counter reaches half of its maximum value or the maximum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
6 RXALGNERPIM	MMC Receive Alignment Error Packet Counter Interrupt Mask Indicates the status of the MMC receive alignment error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxalignmenterror counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
5 RXCRCERPIM	MMC Receive CRC Error Packet Counter Interrupt Mask Indicates the status of the MMC receive CRC error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxcrcerror counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
4 RXMCGPIM	MMC Receive Multicast Good Packet Counter Interrupt Mask Indicates the status of the MMC receive multicast good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxmulticastpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
3 RXBCGPIM	MMC Receive Broadcast Good Packet Counter Interrupt Mask Indicates the status of the MMC receive broadcast good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxbroadcastpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
2 RXGOCTIM	MMC Receive Good Octet Counter Interrupt Mask Indicates the status of the MMC receive good octet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxoctetcount_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
1	MMC Receive Good Bad Octet Counter Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXGBOCTIM	Indicates the status of the MMC receive good bad octet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxoctetcount_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
0 RXGBPKTIM	MMC Receive Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxpacketcount_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled

72.18.61 MMC Transmit Interrupt Mask (MMC_Tx_Interrupt_Mask)

Offset

Register	Offset
MMC_Tx_Interrupt_Mask	710h

Function

Maintains the masks for interrupts that are generated when the transmit statistic counters reach half of their maximum values or their maximum values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				Reserved	Reserved	TXOSIZE...	TXVLANG...	TXPAUSP...	TXEXDEF...	TXGPKTIM	TXGOCTIM	TXCARER...	TXEXCOL...	TXLATCO...	TXDEFPIM
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TXMCOLG...	TXSCOLG...	TXUFLOW...	TXBCGBP...	TXMCGBP...	TXUCGBP...	TX1024T...	TX512T1...	TX256T5...	TX128T2...	TX65T12...	TX64OCT...	TXMCGPIM	TXBCGPIM	TXGBPKT...	TXGBOCT...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 TXOSIZEGPIM	MMC Transmit Oversize Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit oversize good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txoversize_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
24 TXVLANGPIM	MMC Transmit VLAN Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit VLAN good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txvlanpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
23 TXPAUSPIM	MMC Transmit Pause Packet Counter Interrupt Mask Indicates the status of the MMC transmit pause packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txpausepackets counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
22 TXEXDEFPIM	MMC Transmit Excessive Deferral Packet Counter Interrupt Mask Indicates the status of the MMC transmit excessive deferral packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txexcessdef counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
21 TXGPKTIM	MMC Transmit Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit good packet counter interrupt mask.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writing 1 to this field masks the interrupt when the txpacketcount_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
20 TXGOCTIM	MMC Transmit Good Octet Counter Interrupt Mask Indicates the status of the MMC transmit good octet counter interrupt mask. Writing 1 to this field masks the interrupt when the txoctetcount_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
19 TXCARERPIM	MMC Transmit Carrier Error Packet Counter Interrupt Mask Indicates the status of the MMC transmit carrier error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txcarriererror counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
18 TXEXCOLPIM	MMC Transmit Excessive Collision Packet Counter Interrupt Mask Indicates the status of the MMC transmit excessive collision packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txexcesscol counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
17 TXLATCOLPIM	MMC Transmit Late Collision Packet Counter Interrupt Mask Indicates the status of the MMC transmit late collision packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txlatecol counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
16 TXDEFPIM	MMC Transmit Deferred Packet Counter Interrupt Mask Indicates the status of the MMC transmit deferred packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txdeferred counter reaches half of the maximum value or the maximum value. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 TXMCOLGPIM	MMC Transmit Multiple Collision Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit multiple collision good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txmulticol_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
14 TXSCOLGPIM	MMC Transmit Single Collision Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit single collision good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txsinglecol_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
13 TXUFLOWERPIM	MMC Transmit Underflow Error Packet Counter Interrupt Mask Indicates the status of the MMC transmit underflow error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txunderflowerror counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
12 TXBCGBPIM	MMC Transmit Broadcast Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit broadcast good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txbroadcastpackets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
11 TXMCGBPIM	MMC Transmit Multicast Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit multicast good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txmulticastpackets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
10 TXUCGBPIM	MMC Transmit Unicast Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit unicast good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txunicastpackets_gb counter reaches half of its maximum value or the maximum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
9 TX1024TMAXOCTGBPIM	MMC Transmit 1024 To Maximum Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit 1024 to maximum octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the tx1024tomaxoctets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
8 TX512T1023OCTGBPIM	MMC Transmit 512 To 1023 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit 512 to 1023 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the tx512to1023octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
7 TX256T511OCTGBPIM	MMC Transmit 256 To 511 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit 256 to 511 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the tx256to511octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
6 TX128T255OCTGBPIM	MMC Transmit 128 To 255 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit 128 to 255 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the tx128to255octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
5 TX65T127OCTGBPIM	MMC Transmit 65 To 127 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit 65 to 127 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the tx65to127octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
4	MMC Transmit 64-Octet Good Bad Packet Counter Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
TX64OCTGBPIM	Indicates the status of the MMC transmit 64-octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the tx64octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
3 TXMCGPIM	MMC Transmit Multicast Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit multicast good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txmulticastpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
2 TXBCGPIM	MMC Transmit Broadcast Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit broadcast good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txbroadcastpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
1 TXGBPKTIM	MMC Transmit Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txpacketcount_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
0 TXGBOCTIM	MMC Transmit Good Bad Octet Counter Interrupt Mask Indicates the status of the MMC transmit good bad octet counter interrupt mask. Writing 1 to this field masks the interrupt when the txoctetcount_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled

72.18.62 Transmit Octet Count Good Bad (Tx_Octet_Count_Good_Bad)

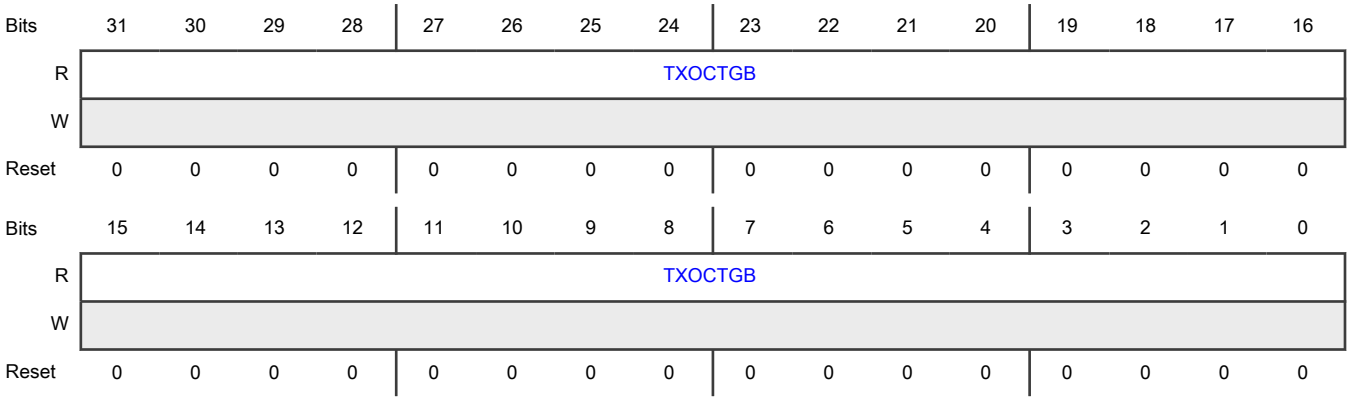
Offset

Register	Offset
Tx_Octet_Count_Good_Bad	714h

Function

Provides the number of bytes that the module transmitted, exclusive of preamble and retried bytes, in good and bad packets.

Diagram



Fields

Field	Function
31-0 TXOCTGB	Transmit Octet Count Good Bad Indicates the number of bytes transmitted, exclusive of preamble and retried bytes, in good and bad packets.

72.18.63 Transmit Packet Count Good Bad (Tx_Packet_Count_Good_Bad)

Offset

Register	Offset
Tx_Packet_Count_Good_Bad	718h

Function

Provides the number of good and bad packets that the module transmitted, exclusive of retried packets.

Diagram

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

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

Fields

Field	Function
31-0 TXPKTGB	Transmit Packet Count Good Bad Indicates the number of good and bad packets transmitted, exclusive of retried packets.

72.18.64 Transmit Broadcast Packets Good (Tx_Broadcast_Packets_Good)

Offset

Register	Offset
Tx_Broadcast_Packets_Good	71Ch

Function

Provides the number of good broadcast packets that the module transmitted.

Diagram

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

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

Fields

Field	Function
31-0 TXBCASTG	Transmit Broadcast Packets Good Indicates the number of good broadcast packets transmitted.

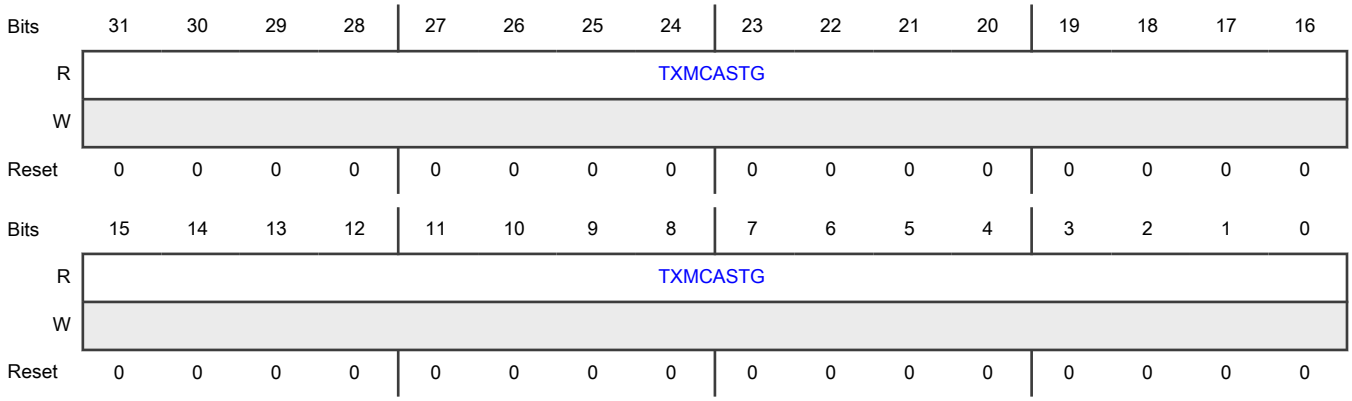
72.18.65 Transmit Multicast Packets Good (Tx_Multicast_Packets_Good)

Offset

Register	Offset
Tx_Multicast_Packets_Good	720h

Function
Provides the number of good multicast packets that the module transmitted.

Diagram



Fields

Field	Function
31-0 TXMCASTG	Transmit Multicast Packets Good Indicates the number of good multicast packets transmitted.

72.18.66 Transmit 64-Octet Packets Good Bad (Tx_64Octets_Packets_Good_Bad)

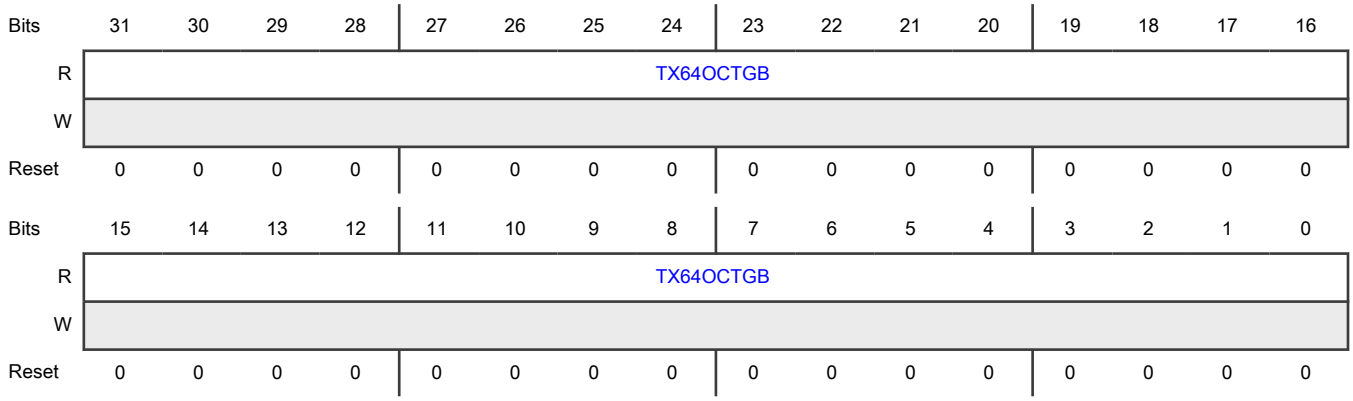
Offset

Register	Offset
Tx_64Octets_Packets_Good_Bad	724h

Function

Provides the number of 64-byte good and bad packets that the module transmitted, exclusive of preamble and retried packets.

Diagram



Fields

Field	Function
31-0 TX64OCTGB	Transmit 64-Octet Packets Good Bad Indicates the number of transmitted 64-byte good and bad packets, exclusive of preamble and retried packets.

72.18.67 Transmit 65 To 127 Octet Packets Good Bad (Tx_65To127Octets_Packets_Good_Bad)

Offset

Register	Offset
Tx_65To127Octets_Packets_Good_Bad	728h

Function

Provides the number of good and bad packets, having length between 65 and 127 (inclusive) bytes, that the module transmitted, exclusive of preamble and retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TX65_127OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX65_127OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TX65_127OCTGB	Transmit 65 To 127 Octet Packets Good Bad Indicates the number of good and bad packets transmitted, having length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried packets.

72.18.68 Transmit 128 To 255 Octet Packets Good Bad (Tx_128To255Octets_Packets_Good_Bad)

Offset

Register	Offset
Tx_128To255Octets_Packets_Good_Bad	72Ch

Function

Provides the number of good and bad packets that the module transmitted, having length between 128 to 255 (inclusive) bytes, exclusive of preamble and retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TX128_255OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX128_255OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TX128_255OCTGB	Transmit 128 To 255 Octet Packets Good Bad Indicates the number of good and bad packets transmitted, having length between 128 and 255 (inclusive) bytes, exclusive of preamble and retried packets.

72.18.69 Transmit 256 To 511 Octet Packets Good Bad (Tx_256To511Octets_Packets_Good_Bad)

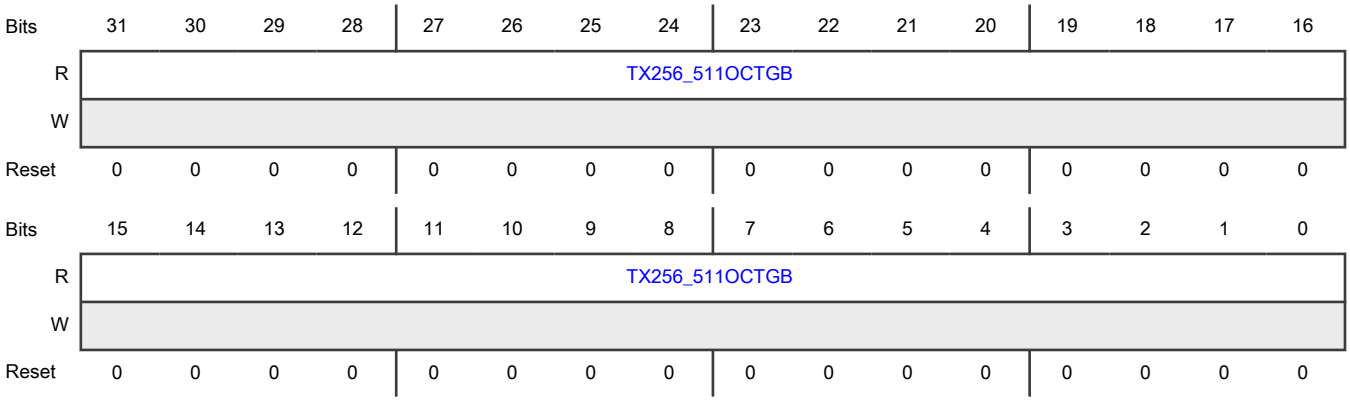
Offset

Register	Offset
Tx_256To511Octets_Packets_Good_Bad	730h

Function

Provides the number of good and bad packets that the module transmitted, having length between 256 to 511 (inclusive) bytes, exclusive of preamble and retried packets.

Diagram



Fields

Field	Function
31-0 TX256_511OCTGB	Transmit 256 To 511 Octet Packets Good Bad Indicates the number of good and bad packets transmitted, having length between 256 and 511 (inclusive) bytes, exclusive of preamble and retried packets.

72.18.70 Transmit 512 To 1023 Octet Packets Good Bad (Tx_512To1023Octets_Packets_Good_Bad)

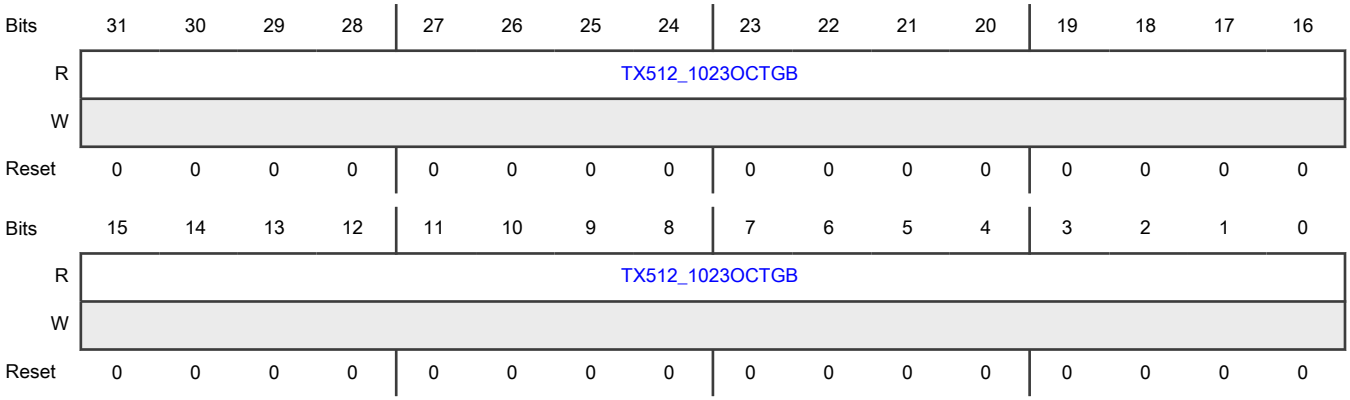
Offset

Register	Offset
Tx_512To1023Octets_Packets_Good_Bad	734h

Function

Provides the number of good and bad packets that the module transmitted, having length between 512 to 1023 (inclusive) bytes, exclusive of preamble and retried packets.

Diagram



Fields

Field	Function
31-0 TX512_1023OCTGB	Transmit 512 To 1023 Octet Packets Good Bad Indicates the number of good and bad packets transmitted, having length between 512 and 1023 (inclusive) bytes, exclusive of preamble and retried packets.

72.18.71 Transmit 1024 To Max Octet Packets Good Bad (Tx_1024ToMaxOctets_Packets_Good_Bad)

Offset

Register	Offset
Tx_1024ToMaxOctets_Packets_Good_Bad	738h

Function

Provides the number of good and bad packets that the module transmitted, having a length between 1024 bytes (inclusive) and the maximum size, exclusive of preamble and retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TX1024_MAXOCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX1024_MAXOCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TX1024_MAXOCTGB	Transmit 1024 To Max Octet Packets Good Bad Indicates the number of good and bad packets transmitted, having a length between 1024 bytes (inclusive) and the maximum size, exclusive of preamble and retried packets.

72.18.72 Transmit Unicast Packets Good Bad (Tx_Unicast_Packets_Good_Bad)

Offset

Register	Offset
Tx_Unicast_Packets_Good_Bad	73Ch

Function

Provides the number of good and bad unicast packets that the module transmitted.

Diagram

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

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

Fields

Field	Function
31-0 TXUCASTGB	Transmit Unicast Packets Good Bad Indicates the number of good and bad unicast packets transmitted.

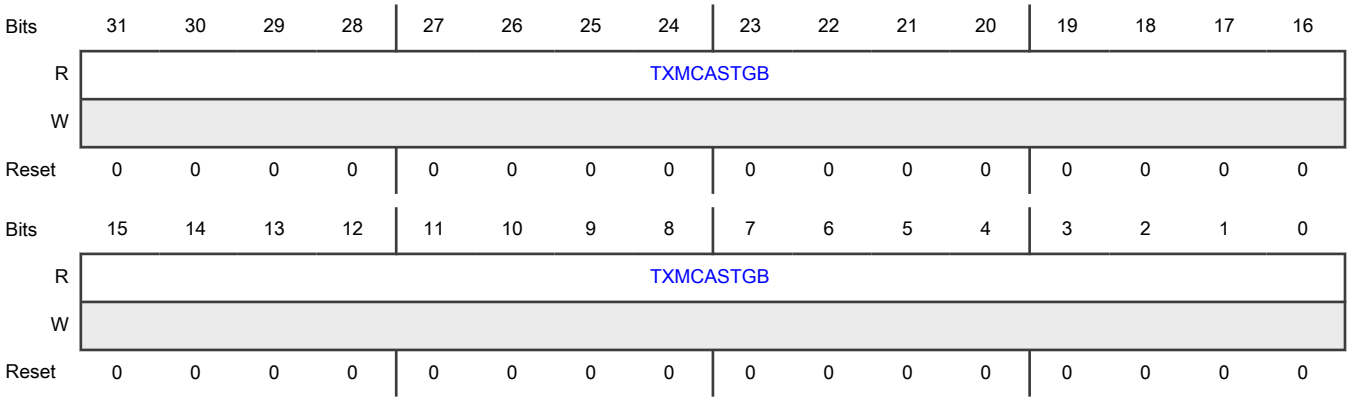
72.18.73 Transmit Multicast Packets Good Bad (Tx_Multicast_Packets_Good_Bad)

Offset

Register	Offset
Tx_Multicast_Packets_Good_Bad	740h

Function
Provides the number of good and bad multicast packets that the module transmitted.

Diagram



Fields

Field	Function
31-0 TXMCASTGB	Transmit Multicast Packets Good Bad Indicates the number of good and bad multicast packets transmitted.

72.18.74 Transmit Broadcast Packets Good Bad (Tx_Broadcast_Packets_Good_Bad)

Offset

Register	Offset
Tx_Broadcast_Packets_Good_Bad	744h

Function

Provides the number of good and bad broadcast packets that the module transmitted.

Diagram

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

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

Fields

Field	Function
31-0 TXBCASTGB	Transmit Broadcast Packets Good Bad Indicates the number of good and bad broadcast packets transmitted.

72.18.75 Transmit Underflow Error Packets (Tx_Underflow_Error_Packets)

Offset

Register	Offset
Tx_Underflow_Error_Packets	748h

Function

Provides the number of packets that the module aborted because of a packet underflow error.

Diagram

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

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

Fields

Field	Function
31-0 TXUNDRFLW	Transmit Underflow Error Packets Indicates the number of packets aborted because of a packet underflow error.

72.18.76 Transmit Single Collision Good Packets (Tx_Single_Collision_Good_Packets)

Offset

Register	Offset
Tx_Single_Collision_Good_Packets	74Ch

Function
Provides the number of packets that the module successfully transmitted after a single collision in Half-Duplex mode.

Diagram

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

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

Fields

Field	Function
31-0 TXSNGLCOLG	Transmit Single Collision Good Packets Indicates the number of successfully transmitted packets after a single collision in Half-Duplex mode.

72.18.77 Transmit Multiple Collision Good Packets (Tx_Multiple_Collision_Good_Packets)

Offset

Register	Offset
Tx_Multiple_Collision_Good_Packets	750h

Function

Provides the number of packets that the module successfully transmitted after multiple collisions in Half-Duplex mode.

Diagram

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

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

Fields

Field	Function
31-0 TXMULTCOLG	Transmit Multiple Collision Good Packets Indicates the number of successfully transmitted packets after multiple collisions in Half-Duplex mode.

72.18.78 Transmit Deferred Packets (Tx_Deferred_Packets)

Offset

Register	Offset
Tx_Deferred_Packets	754h

Function

Provides the number of packets that the module successfully transmitted after a deferral in Half-Duplex mode.

Diagram

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

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

Fields

Field	Function
31-0 TXDEFRD	Transmit Deferred Packets Indicates the number of successfully transmitted packets after a deferral in Half-Duplex mode.

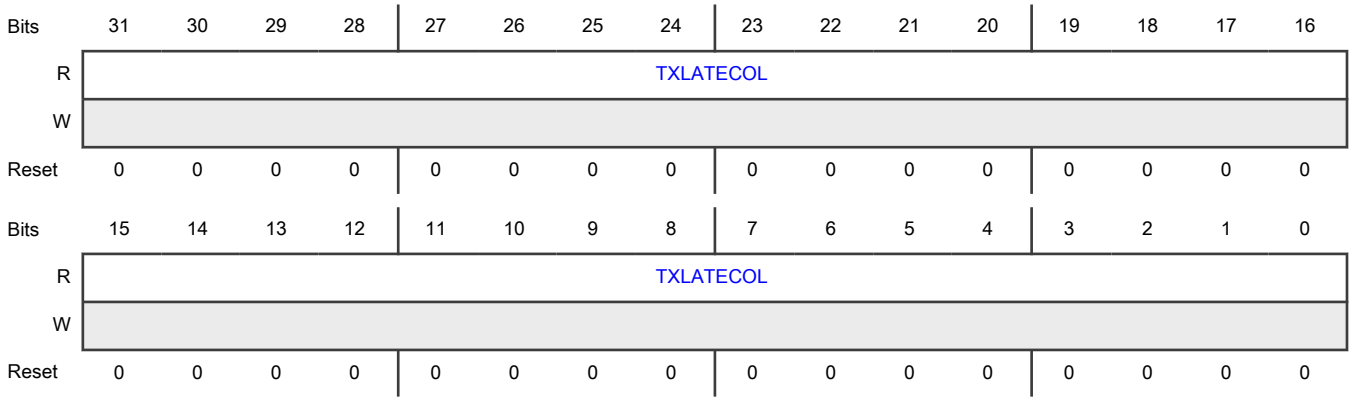
72.18.79 Transmit Late Collision Packets (Tx_Late_Collision_Packets)

Offset

Register	Offset
Tx_Late_Collision_Packets	758h

Function
Provides the number of packets that the module aborted because of a late collision error.

Diagram



Fields

Field	Function
31-0 TXLATECOL	Transmit Late Collision Packets Indicates the number of packets aborted because of a late collision error.

72.18.80 Transmit Excessive Collision Packets (Tx_Excessive_Collision_Packets)

Offset

Register	Offset
Tx_Excessive_Collision_Packets	75Ch

Function

Provides the number of packets that the module aborted because of excessive (16) collision errors.

Diagram

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

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

Fields

Field	Function
31-0 TXEXSCOL	Transmit Excessive Collision Packets Indicates the number of packets aborted because of excessive (16) collision errors.

72.18.81 Transmit Carrier Error Packets (Tx_Carrier_Error_Packets)

Offset

Register	Offset
Tx_Carrier_Error_Packets	760h

Function

Provides the number of packets that the module aborted because of a carrier sense error (such as no carrier or loss of carrier).

Diagram

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

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

Fields

Field	Function
31-0 TXCARR	Transmit Carrier Error Packets Indicates the number of packets aborted because of a carrier sense error (such as no carrier or loss of carrier).

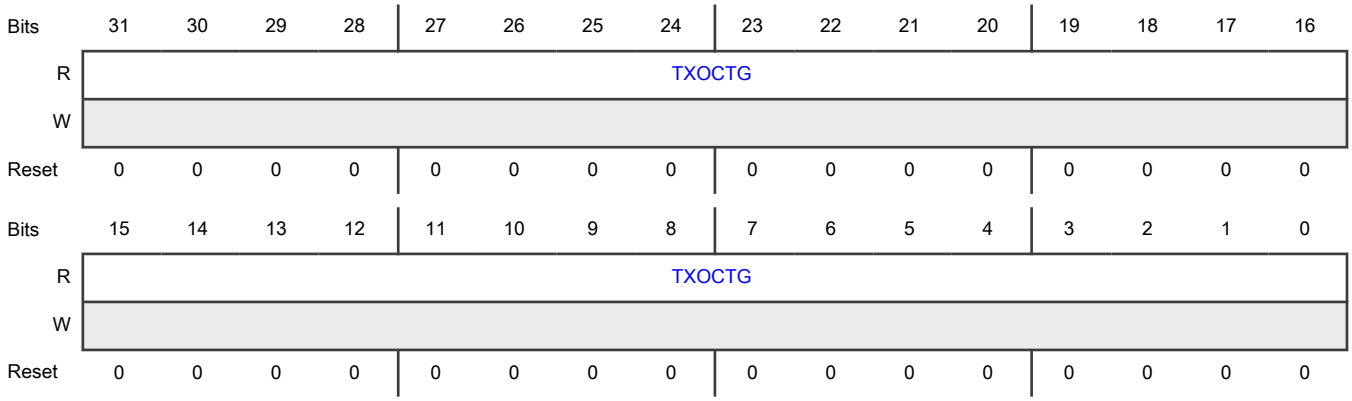
72.18.82 Transmit Octet Count Good (Tx_Octet_Count_Good)

Offset

Register	Offset
Tx_Octet_Count_Good	764h

Function
Provides the number of bytes that the module transmitted, exclusive of preamble, only in good packets.

Diagram



Fields

Field	Function
31-0 TXOCTG	Transmit Octet Count Good Indicates the number of bytes transmitted, exclusive of preamble, only in good packets.

72.18.83 Transmit Packet Count Good (Tx_Packet_Count_Good)

Offset

Register	Offset
Tx_Packet_Count_Good	768h

Function

Provides the number of good packets that the module transmitted.

Diagram

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

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

Fields

Field	Function
31-0 TXPKTG	Transmit Packet Count Good Indicates the number of good packets transmitted.

72.18.84 Transmit Excessive Deferral Error (Tx_Excessive_Deferral_Error)

Offset

Register	Offset
Tx_Excessive_Deferral_Error	76Ch

Function

Provides the number of packets that the module aborted because of an excessive deferral error (deferred for more than two max-sized packet times).

Diagram

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

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

Fields

Field	Function
31-0 TXEXSDEF	Transmit Excessive Deferral Error Indicates the number of packets aborted because of excessive deferral error (deferred for more than two max-sized packet times).

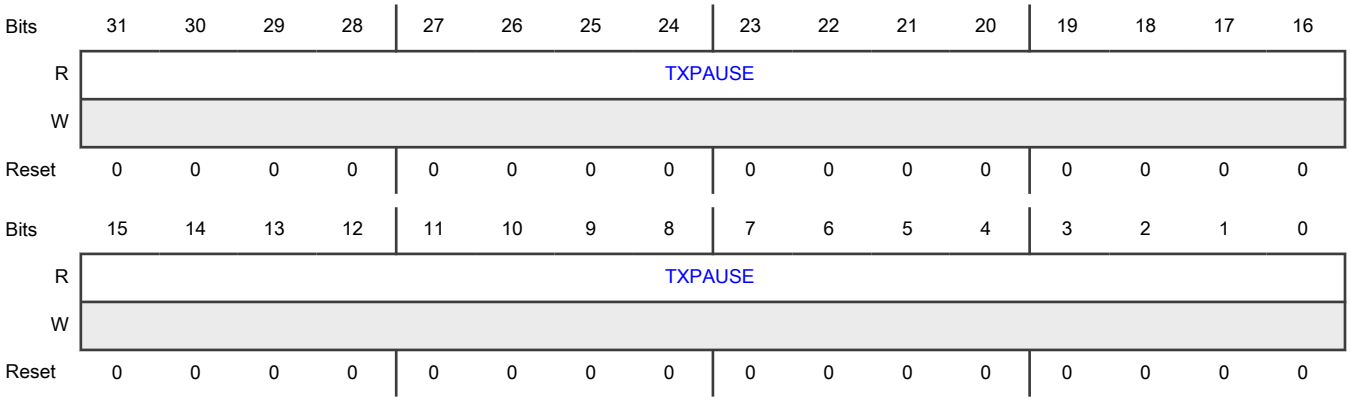
72.18.85 Transmit Pause Packets (Tx_Pause_Packets)

Offset

Register	Offset
Tx_Pause_Packets	770h

Function
Provides the number of good pause packets that the module transmitted.

Diagram



Fields

Field	Function
31-0 TXPAUSE	Transmit Pause Packets Indicates the number of good pause packets transmitted.

72.18.86 Transmit VLAN Packets Good (Tx_VLAN_Packets_Good)

Offset

Register	Offset
Tx_VLAN_Packets_Good	774h

Function

Provides the number of good VLAN packets that the module transmitted.

Diagram

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

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

Fields

Field	Function
31-0 TXVLANG	Transmit VLAN Packets Good Provides the number of good VLAN packets transmitted.

72.18.87 Transmit O Size Packets Good (Tx_OSize_Packets_Good)

Offset

Register	Offset
Tx_OSize_Packets_Good	778h

Function

Provides the number of packets that the module transmitted without errors, and having a length greater than the maximum size, which is 1,518 or 1,522 bytes for VLAN-tagged packets. This size is 2000 bytes if MAC_Configuration[S2KP] = 1.

Diagram

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

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

Fields

Field	Function
31-0 TXOSIZG	Transmit O Size Packets Good Indicates the number of packets transmitted without errors, and having a length greater than the maximum size, which is 1,518 or 1,522 bytes for VLAN-tagged packets. This size is 2000 bytes if MAC_Configuration[S2KP] = 1.

72.18.88 Receive Packets Count Good Bad (Rx_Packets_Count_Good_Bad)

Offset

Register	Offset
Rx_Packets_Count_Good_Bad	780h

Function
Provides the number of good and bad packets that the module received.

Diagram

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

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

Fields

Field	Function
31-0 RXPKTGB	Receive Packets Count Good Bad Indicates the number of good and bad packets received.

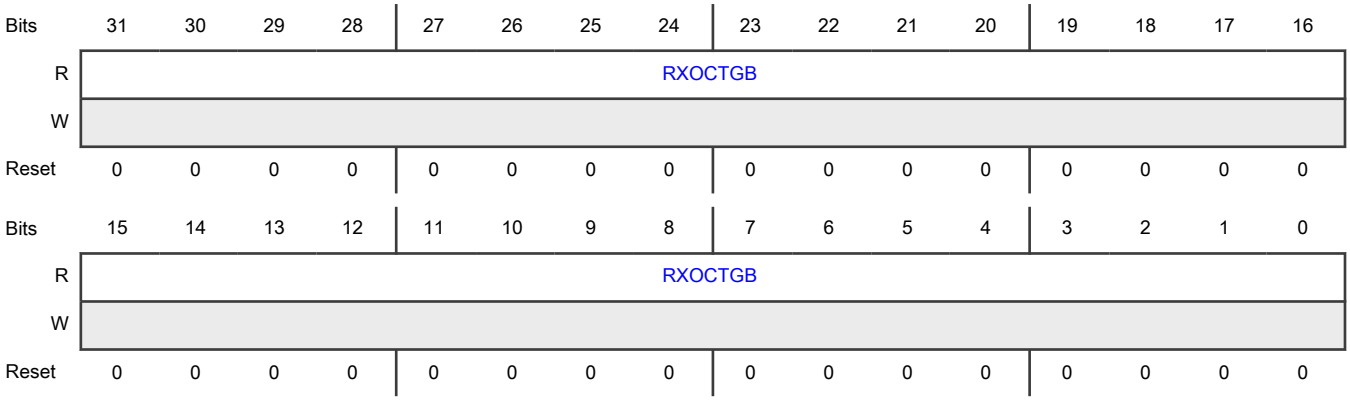
72.18.89 Receive Octet Count Good Bad (Rx_Octet_Count_Good_Bad)

Offset

Register	Offset
Rx_Octet_Count_Good_Bad	784h

Function
Provides the number of bytes that DWC_ther_qos received, exclusive of preamble, in good and bad packets.

Diagram



Fields

Field	Function
31-0 RXOCTGB	Receive Octet Count Good Bad Indicates the number of bytes received, exclusive of preamble, in good and bad packets.

72.18.90 Receive Octet Count Good (Rx_Octet_Count_Good)

Offset

Register	Offset
Rx_Octet_Count_Good	788h

Function
Provides the number of bytes that the module received, exclusive of preamble, only in good packets.

Diagram

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

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

Fields

Field	Function
31-0 RXOCTG	Receive Octet Count Good Indicates the number of bytes received, exclusive of preamble, only in good packets.

72.18.91 Receive Broadcast Packets Good (Rx_Broadcast_Packets_Good)

Offset

Register	Offset
Rx_Broadcast_Packets_Good	78Ch

Function

Provides the number of good broadcast packets that the module received.

Diagram

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

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

Fields

Field	Function
31-0 RXBCASTG	Receive Broadcast Packets Good Indicates the number of good broadcast packets received.

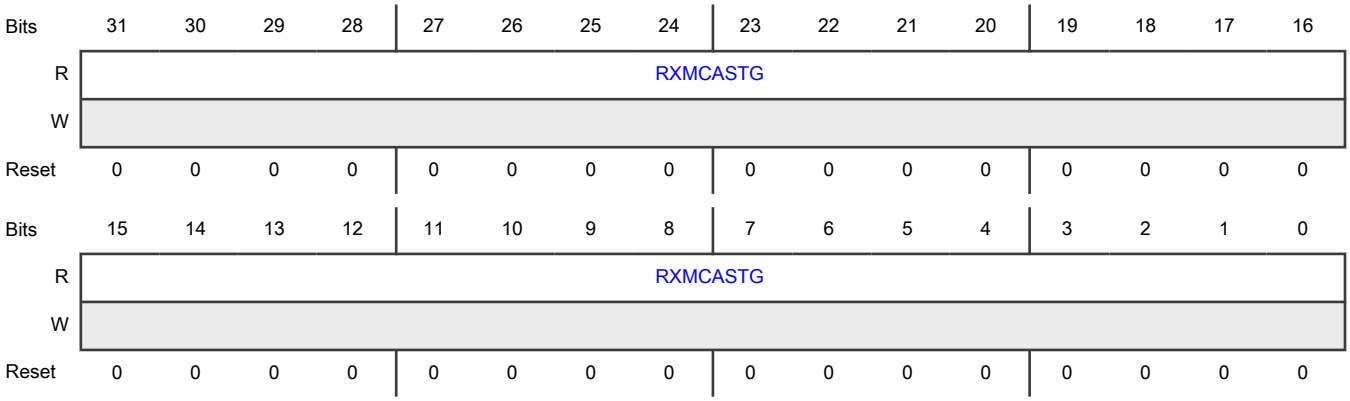
72.18.92 Receive Multicast Packets Good (Rx_Multicast_Packets_Good)

Offset

Register	Offset
Rx_Multicast_Packets_Good	790h

Function
Provides the number of good multicast packets that the module received.

Diagram



Fields

Field	Function
31-0 RXMCASTG	Receive Multicast Packets Good Indicates the number of good multicast packets received.

72.18.93 Receive CRC Error Packets (Rx_CRC_Error_Packets)

Offset

Register	Offset
Rx_CRC_Error_Packets	794h

Function

Provides the number of packets that the module received with a CRC error.

Diagram

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

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

Fields

Field	Function
31-0 RXCRCERR	Receive CRC Error Packets Indicates the number of packets received with a CRC error.

72.18.94 Receive Alignment Error Packets (Rx_Alignment_Error_Packets)

Offset

Register	Offset
Rx_Alignment_Error_Packets	798h

Function

Provides the number of packets that the module received with an alignment (dribble) error. It is valid only in 10/100 mode.

Diagram

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

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

Fields

Field	Function
31-0 RXALGNERR	Receive Alignment Error Packets Indicates the number of packets received with alignment (dribble) error. It is valid only in 10/100 mode.

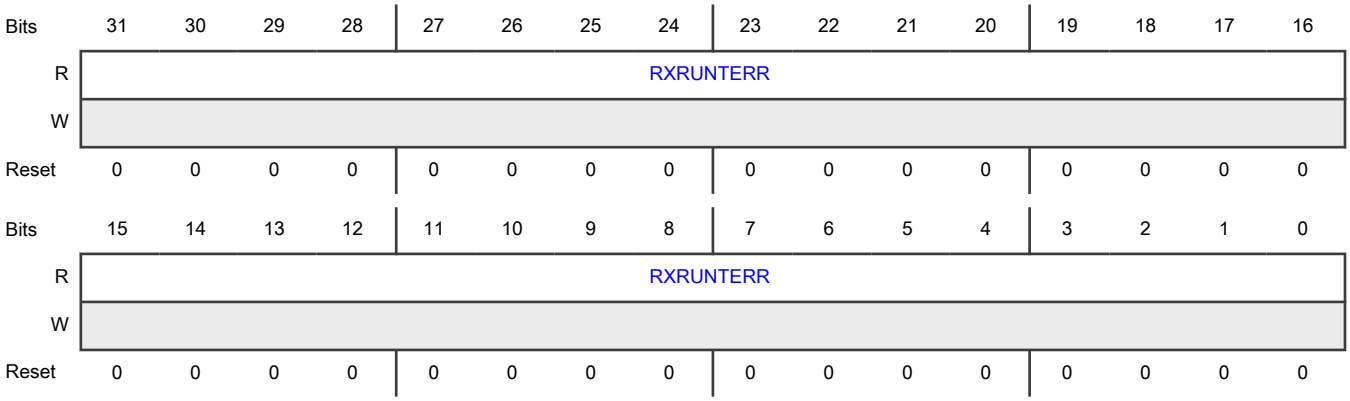
72.18.95 Receive Runt Error Packets (Rx_Runt_Error_Packets)

Offset

Register	Offset
Rx_Runt_Error_Packets	79Ch

Function
Provides the number of runt packets, which have a length less than 64 bytes and a CRC error, that the module received.

Diagram



Fields

Field	Function
31-0 RXRUNTERR	Receive Runt Error Packets Indicates the number of runt packets received.

72.18.96 Receive Jabber Error Packets (Rx_Jabber_Error_Packets)

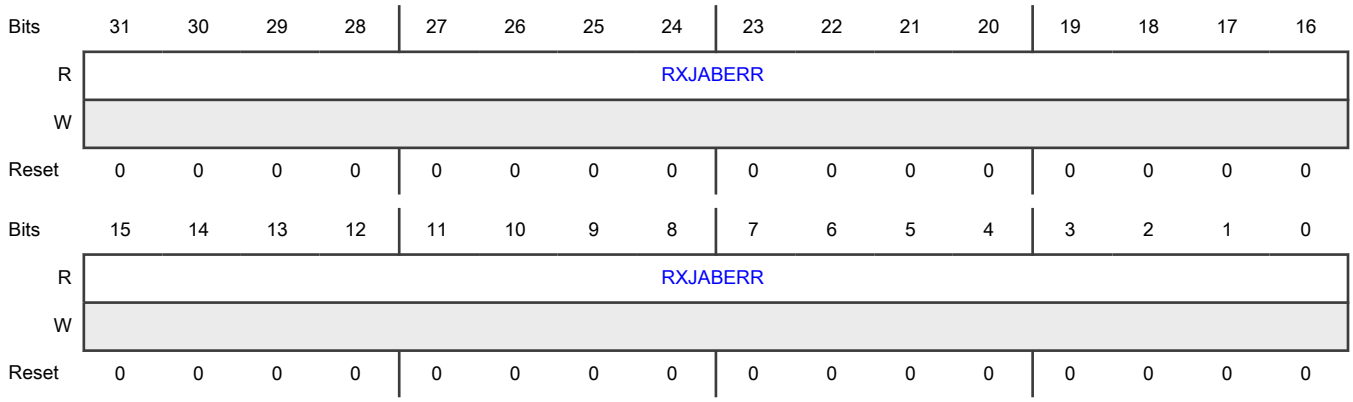
Offset

Register	Offset
Rx_Jabber_Error_Packets	7A0h

Function

Provides the number of giant packets that the module received, having a length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN-tagged packets), and with a CRC error. If Jumbo Packet mode is enabled, packets of length greater than 9,018 bytes (9,022 bytes for VLAN-tagged packets) are considered as giant packets.

Diagram



Fields

Field	Function
31-0 RXJABERR	Receive Jabber Error Packets Indicates the number of giant packets received, having a length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN-tagged packets), and with a CRC error. If Jumbo Packet mode is enabled, packets of length greater than 9,018 bytes (9,022 bytes for VLAN-tagged packets) are considered as giant packets.

72.18.97 Receive Undersize Packets Good (Rx_Undersize_Packets_Good)

Offset

Register	Offset
Rx_Undersize_Packets_Good	7A4h

Function

Provides the number of packets that the module received, having a length less than 64 bytes, but without any errors.

Diagram

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

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

Fields

Field	Function
31-0 RXUNDERSZG	Receive Undersize Packets Good Indicates the number of packets received, having a length less than 64 bytes, but without any errors.

72.18.98 Receive Oversize Packets Good (Rx_Oversize_Packets_Good)

Offset

Register	Offset
Rx_Oversize_Packets_Good	7A8h

Function

Provides the number of packets that the module received without any errors, having a length greater than the maximum size (1,518 bytes or 1,522 bytes for VLAN-tagged packets; 2000 bytes if MAC_Configuration[S2KP] = 1).

Diagram

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

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

Fields

Field	Function
31-0 RXOVERSZG	Receive Oversize Packets Good Indicates the number of packets received without errors, with a length greater than the maximum size (1,518 bytes or 1,522 bytes for VLAN-tagged packets; 2000 bytes if MAC_Configuration[S2KP] = 1).

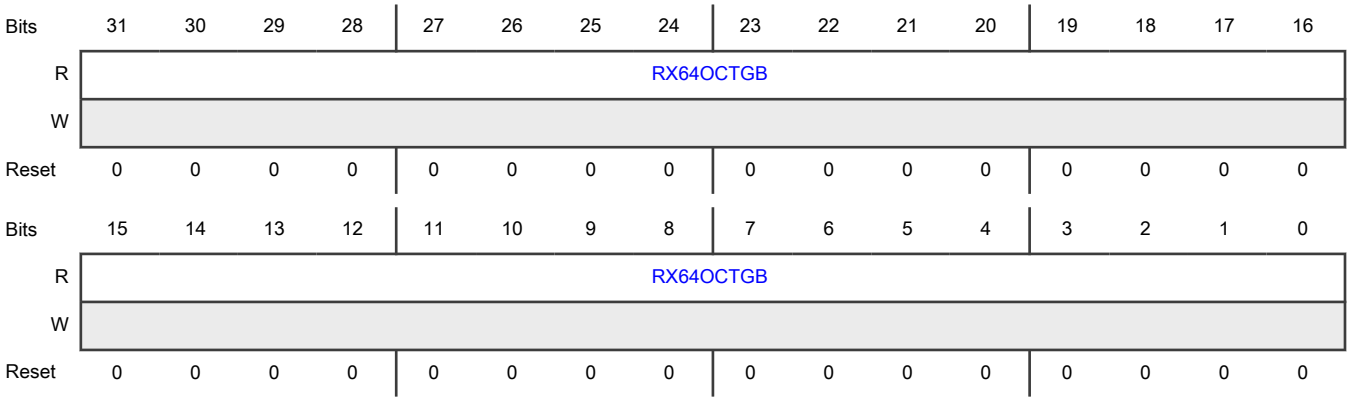
72.18.99 Receive 64 Octets Packets Good Bad (Rx_64Octets_Packets_Good_Bad)

Offset

Register	Offset
Rx_64Octets_Packets_Good_Bad	7ACh

Function
Provides the number of 64-byte good and bad packets that the module received, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX64OCTGB	Receive 64 Octets Packets Good Bad Provides the number of 64-byte good and bad packets that the module received, exclusive of the preamble.

72.18.100 Receive 65-127 Octets Packets Good Bad (Rx_65To127Octets_Packets_Good_Bad)

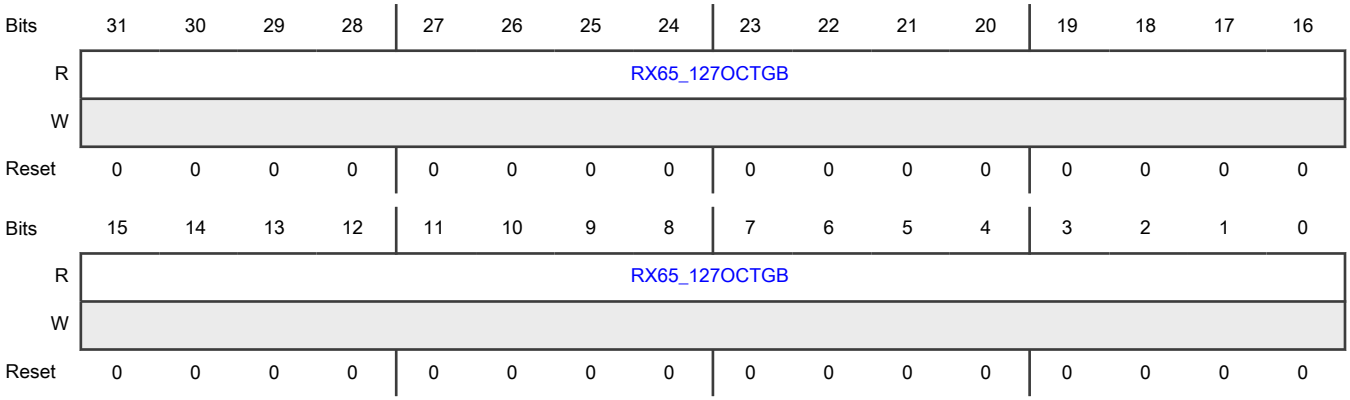
Offset

Register	Offset
Rx_65To127Octets_Packets_Good_Bad	7B0h

Function

Provides the number of good and bad packets that the module received, ranging between 65 and 127 (inclusive) bytes, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX65_127OCTGB	Receive 65-127 Octets Packets Good Bad Indicates the number of good and bad packets received, ranging between 65 and 127 (inclusive) bytes, exclusive of the preamble.

72.18.101 Receive 128-255 Octets Packets Good Bad (Rx_128To255Octets_Packets_Good_Bad)

Offset

Register	Offset
Rx_128To255Octets_Packets_Good_Bad	7B4h

Function

Provides the number of good and bad packets that the module received, with packet length between 128 and 255 (inclusive) bytes, exclusive of the preamble.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RX128_255OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RX128_255OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RX128_255OCTGB	Receive 128-255 Octets Packets Good Bad Indicates the number of good and bad packets received, with packet length between 128 and 255 (inclusive) bytes, exclusive of the preamble.

72.18.102 Receive 256-511 Octets Packets Good Bad (Rx_256To511Octets_Packets_Good_Bad)

Offset

Register	Offset
Rx_256To511Octets_Packets_Good_Bad	7B8h

Function

Provides the number of good and bad packets that the module received, having a packet length between 256 and 511 (inclusive) bytes, exclusive of the preamble.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RX256_511OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RX256_511OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RX256_511OCTGB	Receive 256-511 Octets Packets Good Bad Indicates the number of good and bad packets received, having a packet length between 256 and 511 (inclusive) bytes, exclusive of the preamble.

72.18.103 Receive 512-1023 Octets Packets Good Bad (Rx_512To1023Octets_Packets_Good_Bad)

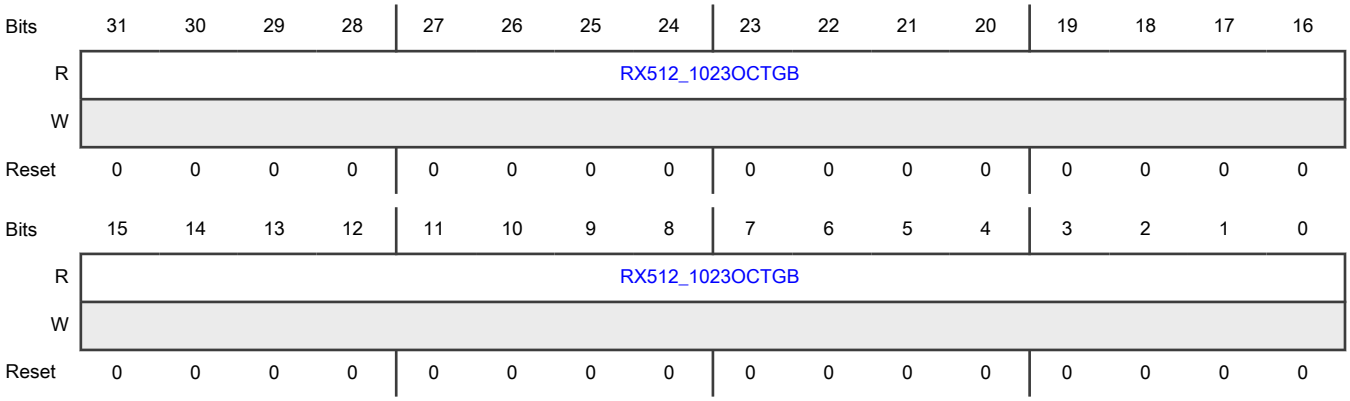
Offset

Register	Offset
Rx_512To1023Octets_Packets_Good_Bad	7BCh

Function

Provides the number of good and bad packets that the module received, having a packet length between 512 and 1023 (inclusive) bytes, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX512_1023OCTGB	Receive 512-1023 Octets Packets Good Bad Indicates the number of good and bad packets received, having a packet length between 512 and 1023 (inclusive) bytes, exclusive of the preamble.

72.18.104 Receive 1024 To Max Octets Good Bad (Rx_1024ToMaxOctets_Packets_Good_Bad)

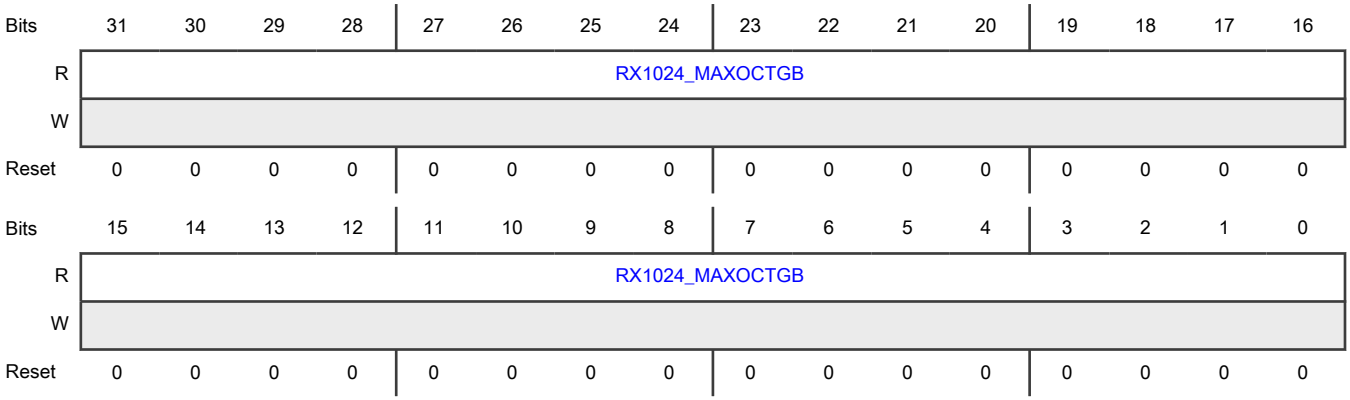
Offset

Register	Offset
Rx_1024ToMaxOctets_Packets_Good_Bad	7C0h

Function

Provides the number of good and bad packets that the module received, having an inclusive packet length between 1024 bytes and the maximum byte size, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX1024_MAXOCTGB	Receive 1024-Max Octets Good Bad Indicates the number of good and bad packets received with length between 1024 and the maximum size (inclusive) bytes, exclusive of the preamble.

72.18.105 Receive Unicast Packets Good (Rx_Unicast_Packets_Good)

Offset

Register	Offset
Rx_Unicast_Packets_Good	7C4h

Function

Provides the number of good unicast packets that the module received.

Diagram

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

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

Fields

Field	Function
31-0 RXUCASTG	Receive Unicast Packets Good Indicates the number of good unicast packets received.

72.18.106 Receive Length Error Packets (Rx_Length_Error_Packets)

Offset

Register	Offset
Rx_Length_Error_Packets	7C8h

Function

Provides the number of packets that the module received with a length error (the Length/Type (LT) field of the RDES3 normal descriptor not equal to the packet size) for all the packets with a valid length field. For more information on the LT field, see the "RDES3 normal descriptor (write-back format)" section.

Diagram

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

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

Fields

Field	Function
31-0 RXLENERR	Receive Length Error Packets Indicates the number of packets received with a length error (the Length/Type (LT) field of the RDES3 normal descriptor not equal to the packet size) for all the packets with a valid length field. For more information on the LT field, see the "RDES3 normal descriptor (write-back format)" section.

72.18.107 Receive Out of Range Type Packet (Rx_Out_Of_Range_Type_Packets)

Offset

Register	Offset
Rx_Out_Of_Range_Type_Packets	7CCh

Function

Provides the number of packets that the module received, with the length field not equal to the valid packet size (greater than 1,500 but less than 1,536).

Diagram

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

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

Fields

Field	Function
31-0 RXOUTOFRNG	Receive Out of Range Type Packet Indicates the number of packets received with length field not equal to the valid packet size (greater than 1,500 but less than 1,536).

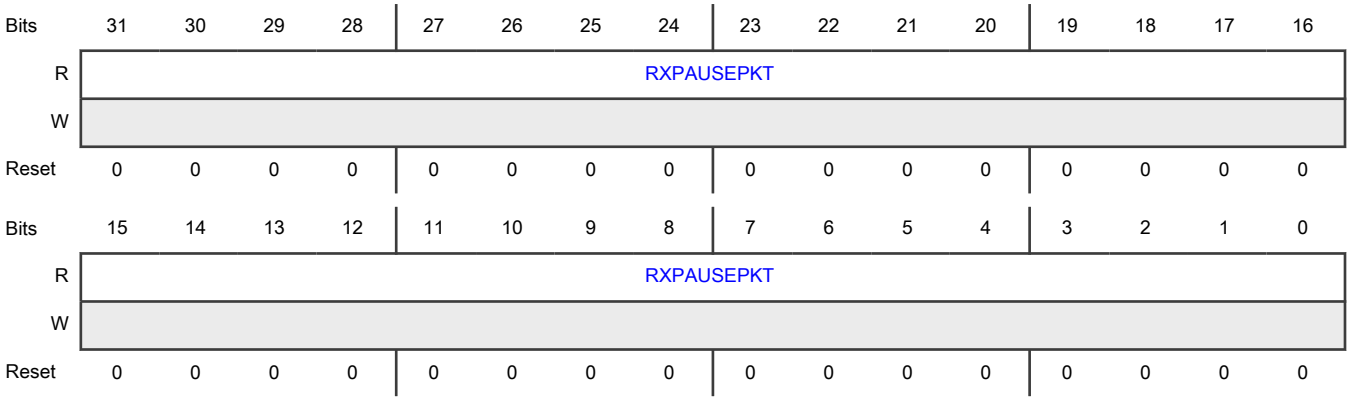
72.18.108 Receive Pause Packets (Rx_Pause_Packets)

Offset

Register	Offset
Rx_Pause_Packets	7D0h

Function
Provides the number of good and valid pause packets that the module received.

Diagram



Fields

Field	Function
31-0 RXPAUSEPKT	Receive Pause Packets Indicates the number of good and valid pause packets received.

72.18.109 Receive FIFO Overflow Packets (Rx_FIFO_Overflow_Packets)

Offset

Register	Offset
Rx_FIFO_Overflow_Packets	7D4h

Function
Provides the number of missed received packets because of FIFO overflow in the module.

Diagram

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

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

Fields

Field	Function
31-0 RXFIFOOVFL	Receive FIFO Overflow Packets Indicates the number of missed received packets because of FIFO overflow.

72.18.110 Receive VLAN Packets Good Bad (Rx_VLAN_Packets_Good_Bad)

Offset

Register	Offset
Rx_VLAN_Packets_Good_Bad	7D8h

Function

Provides the number of good and bad VLAN packets that the module received.

Diagram

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

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

Fields

Field	Function
31-0 RXVLANPKTGB	Receive VLAN Packets Good Bad Indicates the number of good and bad VLAN packets received.

72.18.111 Receive Watchdog Error Packets (Rx_Watchdog_Error_Packets)

Offset

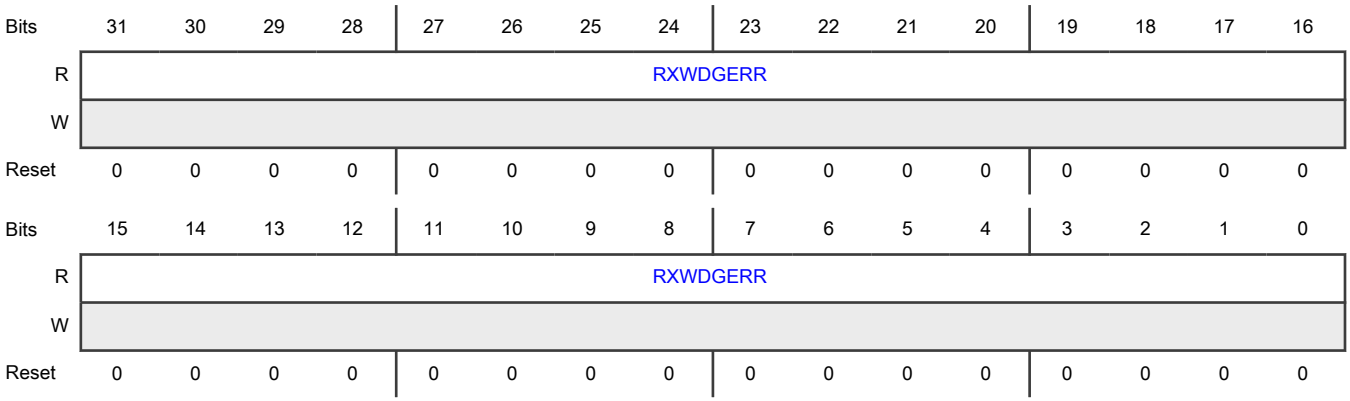
Register	Offset
Rx_Watchdog_Error_Packets	7DCh

Function

Provides the number of packets that the module received with a watchdog timeout error. Packets with a data load larger than the following are received:

- 2,048 bytes when MAC_Configuration[JE] = 0 and MAC_Configuration[WD] = 0
- 10,240 bytes when MAC_Configuration[JE] = 1 and MAC_Configuration[WD] = 0
- 16,384 bytes when MAC_Configuration[WD] = 1 or the value programmed in MAC Watchdog Timeout (MAC_Watchdog_Timeout)

Diagram



Fields

Field	Function
31-0 RXWDGERR	Receive Watchdog Error Packets Indicates the number of packets received with a watchdog timeout error. Packets with a data load larger than the following are received:

Table continues on the next page...

Field	Function
	• 2,048 bytes when MAC_Configuration[JE] = 0 and MAC_Configuration[WD] = 0 • 10,240 bytes when MAC_Configuration[JE] = 1 and MAC_Configuration[WD] = 0 • 16,384 bytes when MAC_Configuration[WD] = 1 or the value programmed in MAC Watchdog Timeout (MAC_Watchdog_Timeout)

72.18.112 Receive Receive Error Packets (Rx_Receive_Error_Packets)

Offset

Register	Offset
Rx_Receive_Error_Packets	7E0h

Function

Provides the number of packets that the module received with a receive or packet extension error on GMII or MII.

Diagram

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

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

Fields

Field	Function
31-0 RXRCVERR	Receive Receive Error Packets Indicates the number of packets received with a receive or packet extension error on GMII or MII.

72.18.113 Receive Control Packets Good (Rx_Control_Packets_Good)

Offset

Register	Offset
Rx_Control_Packets_Good	7E4h

Function

Provides the number of good control packets that the module received.

Diagram

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

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

Fields

Field	Function
31-0 RXCTRLG	Receive Control Packets Good Indicates the number of good control packets received.

72.18.114 MMC Transmit FPE Fragment Counter Interrupt Status (MMC_FPE_Tx_Interrupt)

Offset

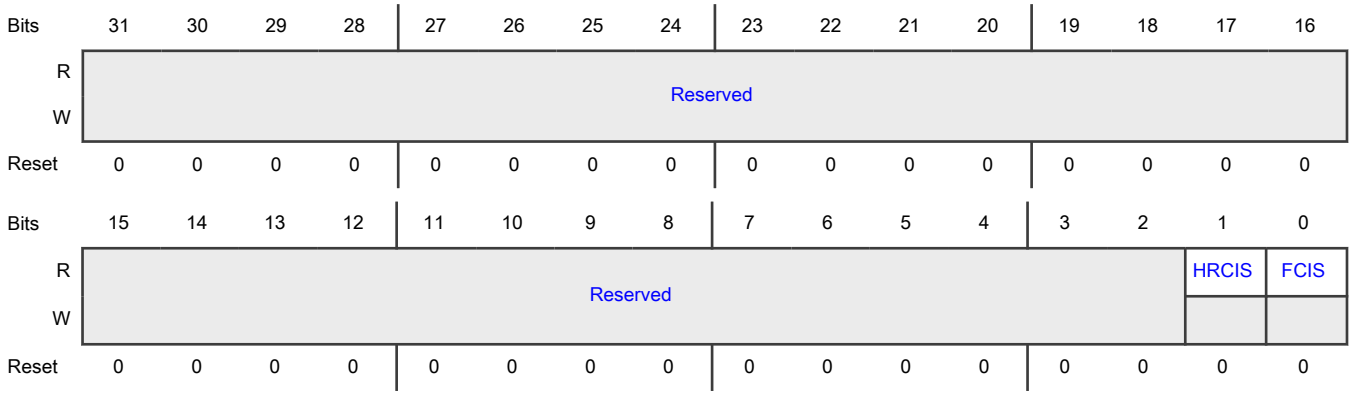
Register	Offset
MMC_FPE_Tx_Interrupt	8A0h

Function

Maintains the interrupts generated from all the FPE-related transmit statistic counters when they reach half of their maximum values (8000_0000h for a 32-bit counter and 8000h for a 16-bit counter), and when they cross their maximum values (FFFF_FFFFh for a 32-bit counter and FFFFh for a 16-bit counter). When MMC_Control[CNTSTOPRO] = 1, the interrupts are set but the counter remains at all-ones.

An interrupt bit of this register becomes 0 when the respective MMC counter that caused the interrupt is read. The least significant byte lane (bits [7:0]) of the respective counter must be read to clear the interrupt bit.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 HRCIS	MMC Transmit Hold Request Counter Interrupt Status Indicates whether the MMC Transmit hold request counter interrupt status is detected. This field: • Becomes 1 when the Tx_Hold_Req_Cntr counter reaches half of the maximum value or the maximum value. • Exists when any one of the receive or transmit MMC counters is enabled during FPE with AV_EST-enabled configuration. Access restrictions apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
0 FCIS	MMC Transmit FPE Fragment Counter Interrupt Status Indicates whether the MMC Transmit FPE fragment counter interrupt status is detected. This field: • Becomes 1 when the Tx_FPE_Fragment_Cntr counter reaches half of the maximum value or the maximum value. • Exists when any one of the receive or transmit MMC counters is enabled during FPE-enabled configuration. Access restrictions apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

72.18.115 MMC FPE Transmit Interrupt Mask (MMC_FPE_Tx_Interrupt_Mask)

Offset

Register	Offset
MMC_FPE_Tx_Interrupt_Mask	8A4h

Function

Maintains the masks for interrupts generated from all FPE-related transmit statistics counters. MMC Receive Interrupt Mask (MMC_Rx_Interrupt_Mask) maintains the masks for the interrupts generated when FPE-related receive statistic counters reach half of their maximum value or the maximum value.

Diagram

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

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

Fields

Field	Function
31-2 —	Reserved
1 HRCIM	MMC Transmit Hold Request Counter Interrupt Mask Indicates the status of the MMC transmit hold request counter interrupt mask. Writing 1 to this field masks the interrupt when the Tx_Hold_Req_Cntr counter reaches half of its maximum value or the maximum value. The field exists when any one of the receive, transmit, or MMC counters is enabled during FPE with AV_EST-enabled configuration. 0b - Disabled 1b - Enabled
0 FCIM	MMC Transmit Fragment Counter Interrupt Mask Indicates the status of the MMC transmit fragment counter interrupt mask.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writing 1 to this field masks the interrupt when the Tx_FPE_Fragment_Cntr counter reaches half of its maximum value or the maximum value. The field exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. 0b - Disabled 1b - Enabled

72.18.116 Transmit FPE Fragment Counter (MMC_Tx_FPE_Fragment_Cntr)

Offset

Register	Offset
MMC_Tx_FPE_Fragment_Cntr	8A8h

Function

Provides the number of additional mPackets transmitted because of preemption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXFFC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXFFC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXFFC	Transmit FPE Fragment Counter Indicates the number of additional mPackets transmitted because of preemption. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration.

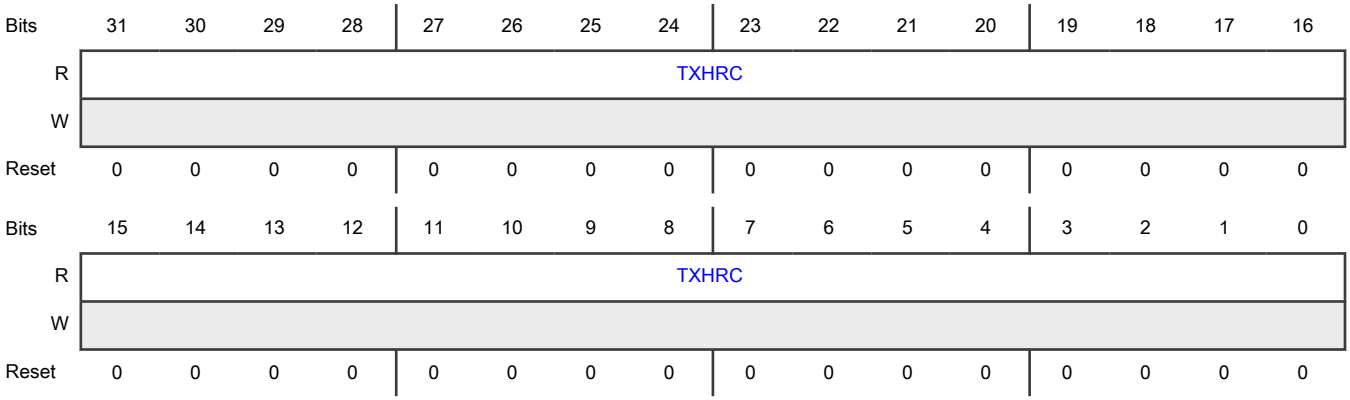
72.18.117 Transmit Hold Request Counter (MMC_Tx_Hold_Req_Cntr)

Offset

Register	Offset
MMC_Tx_Hold_Req_Cntr	8ACh

Function
Provides the count of times that a hold request is given to MAC.

Diagram



Fields

Field	Function
31-0 TXHRC	Transmit Hold Request Counter Indicates the count of times that a hold request is given to MAC. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE with AV_EST-enabled configuration.

72.18.118 MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt)

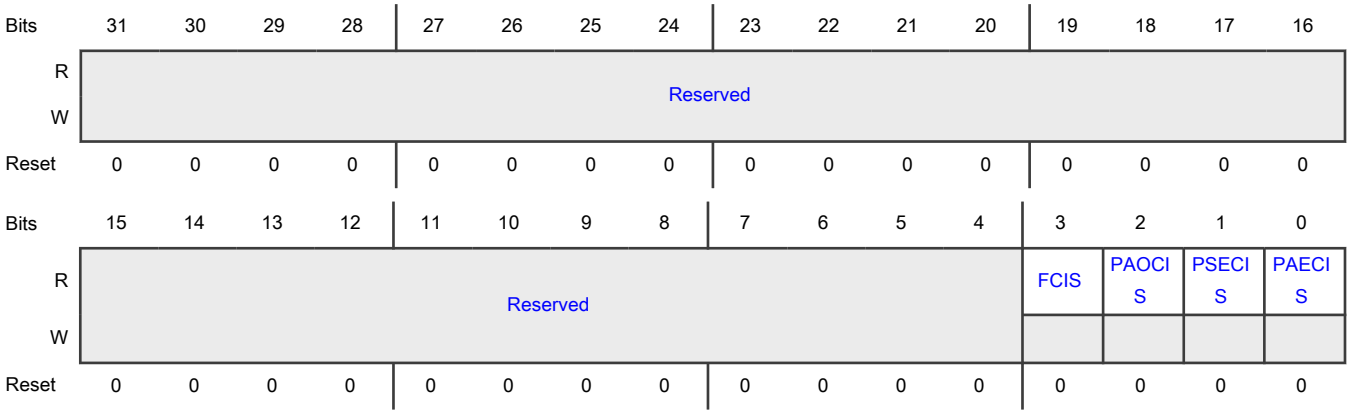
Offset

Register	Offset
MMC_FPE_Rx_Interrupt	8C0h

Function
Maintains the interrupts generated from all the FPE-related receive statistic counters, when these transmit statistic counters reach half of their maximum values (8000_0000h for a 32-bit counter and 8000h for a 16-bit counter), and when they cross their maximum values (FFFF_FFFFh for a 32-bit counter and FFFFh for 16-bit counter). When MMC_Control[CNTSTOPRO] = 1, the interrupts are set but the counter remains at all-ones. An interrupt bit of this register becomes 0 when the respective

MMC counter that caused the interrupt is read. The least significant byte lane (bits [7:0]) of the respective counter must be read to clear the interrupt bit.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 FCIS	MMC Receive FPE Fragment Counter Interrupt Status Indicates whether the MMC receive FPE fragment counter interrupt status is detected. This field becomes 1 when the Rx_FPE_Fragment_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. Access restrictions apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
2 PAOCIS	MMC Receive Packet Assembly OK Counter Interrupt Status Indicates whether the MMC receive packet assembly OK counter interrupt status is detected. This field becomes 1 when the Rx_Packet_Assemble_Ok_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. Access restrictions apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 PSECIS	MMC Receive, Transmit, Packet SMD Error Counter Interrupt Status Indicates whether the status of the MMC receive packet SMD error counter interrupt is detected. This field becomes 1 when the Rx_Packet_SMD_Err_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. Access restrictions apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
0 PAECIS	MMC Receive, transmit, Packet Assembly Error Counter Interrupt Status Indicates whether the status of the MMC receive packet assembly error counter interrupt is detected. This field becomes 1 when the Rx_Packet_Assemble_Err_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during the FPE-enabled configuration. Access restrictions apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

72.18.119 MMC FPE Receive Interrupt Mask (MMC_FPE_Rx_Interrupt_Mask)

Offset

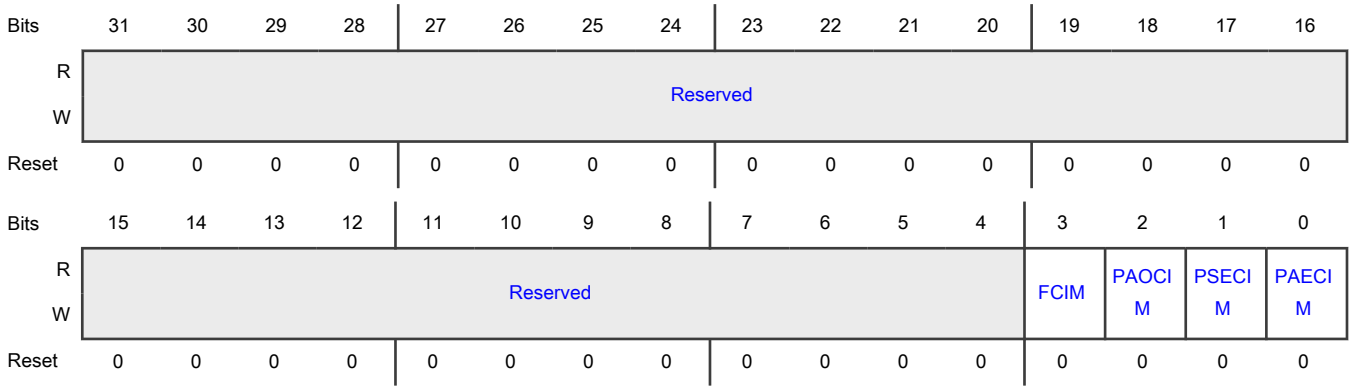
Register	Offset
MMC_FPE_Rx_Interrupt_Mask	8C4h

Function

Maintains the masks for interrupts generated from all FPE-related receive statistic counters.

This register maintains the masks for the interrupts generated when FPE-related receive statistic counters reach half of their maximum value or the maximum value.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 FCIM	MMC Receive FPE Fragment Counter Interrupt Mask Indicates the status of the MMC receive FPE fragment counter interrupt mask. Writing 1 to this field masks the interrupt when the Tx_FPE_Fragment_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. 0b - Disabled 1b - Enabled
2 PAOCIM	MMC Receive Packet Assembly OK Counter Interrupt Mask Indicates the status of the MMC receive packet assembly OK counter interrupt mask. Writing 1 to this field masks the interrupt when the Rx_Packet_Assemble_Ok_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. 0b - Disabled 1b - Enabled
1 PSECIM	MMC Receive Packet SMD Error Counter Interrupt Mask Indicates the status of the MMC receive packet SMD error counter interrupt mask. Writing 1 to this field masks the interrupt when the Rx_Packet_SMD_Err_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
0 PAECIM	MMC receive Packet Assembly Error Counter Interrupt Mask Indicates the status of the MMC receive packet assembly error counter interrupt mask. Writing 1 to this field masks the interrupt when the Rx_Packet_Assemble_Err_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. 0b - Disabled 1b - Enabled

72.18.120 MMC Receive Packet Assembly Error Counter (MMC_Rx_Packet_Assembly_Err_Cntr)

Offset

Register	Offset
MMC_Rx_Packet_Assembly_Err_Cntr	8C8h

Function

Provides the number of MAC frames having reassembly errors on the receiver arising out of mismatch in the fragment count value.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PAEC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PAEC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PAEC	Packet Assembly Error Counter Indicates the number of MAC frames having reassembly errors on the receiver arising out of mismatch in the fragment count value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration.

72.18.121 MMC Receive Packet SMD Error Counter (MMC_Rx_Packet_SMD_Err_Cntr)

Offset

Register	Offset
MMC_Rx_Packet_SMD_Err_Cntr	8CCh

Function

Provides the number of received MAC frames rejected because of an unknown SMD value and MAC frame fragments rejected because of arriving with SMD-C when there was no preceding preempted frame.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PSEC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PSEC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PSEC	Packet SMD Error Counter Indicates the number of MAC frames rejected because of an unknown SMD value and MAC frame fragments rejected because of arriving with SMD-C when there was no preceding preempted frame. Exists when at least one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration.

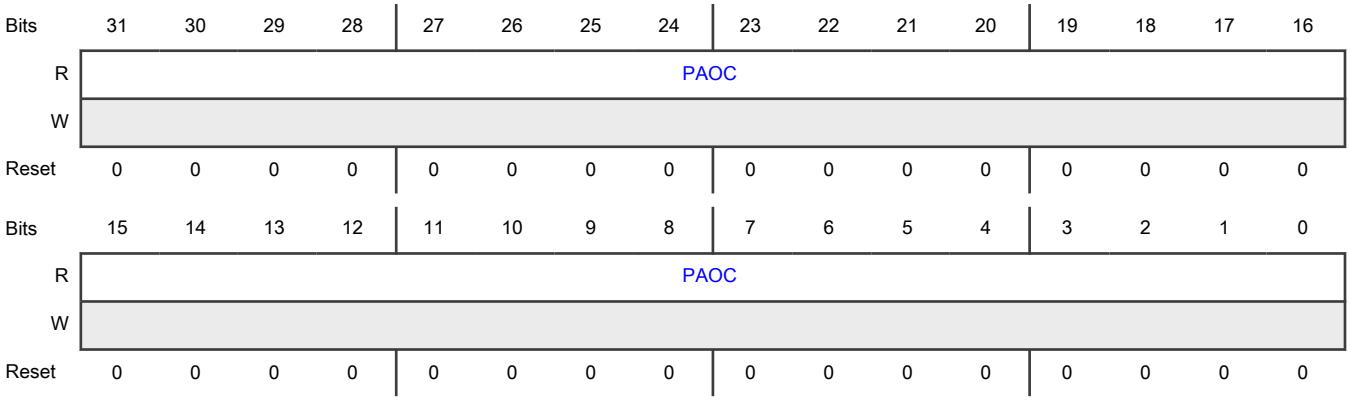
72.18.122 MMC Receive Packet Assembly OK Counter (MMC_Rx_Packet_Assembly_OK_Cntr)

Offset

Register	Offset
MMC_Rx_Packet_Assembly_OK_Cntr	8D0h

Function
Provides the number of MAC frames that were successfully reassembled and delivered to MAC.

Diagram



Fields

Field	Function
31-0 PAOC	Packet Assembly OK Counter Indicates the number of MAC frames that were successfully reassembled and delivered to MAC. Exists when at least one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration.

72.18.123 MMC Receive FPE Fragment Counter (MMC_Rx_FPE_Fragment_Cntr)

Offset

Register	Offset
MMC_Rx_FPE_Fragment_Cntr	8D4h

Function
Provides the number of additional mPackets received because of preemption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FFC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FFC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 FFC	FPE Fragment Counter Indicates the number of additional mPackets received because of preemption. Exists when at least one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration.

72.18.124 MAC Layer 3 Layer 4 Control 0 (MAC_L3_L4_Control0)

Offset

Register	Offset
MAC_L3_L4_Control0	900h

Function
Controls the filter 0 operations of the layer 3 and layer 4 protocols.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			DMCHEN0	Reserved			DMCHN0	Reserved		L4DPIM0	L4DPM0	L4SPIM0	L4SPM0	Reserved	L4PEN0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L3HDBM0				L3HSBM0						L3DAIM0	L3DAM0	L3SAIM0	L3SAM0	Reserved	L3PEN0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN0	DMA Channel Select Enable Indicates the status of the DMA channel number. • If this field is 1, it enables the selection of the DMA channel number for the packet that the L3_L4 filter passes. The DMCHN fields indicate the DMA channel number. • If this field is 0, the filter does not decide the DMA channel number. 0b - Disabled 1b - Enabled
27-25 —	Reserved
24 DMCHN0	DMA Channel Number Indicates the DMA channel number. If the value of the DMCHEN fields is 1, this field selects the DMA channel number to which the packet that this filter passes is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM0	Layer 4 Destination Port Inverse Match Enable Indicates the status of layer 4 destination port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4DP0] is enabled for perfect matching. This field is valid and applicable only when the L4DPM0 field is 1. 0b - Disabled 1b - Enabled
20 L4DPM0	Layer 4 Destination Port Match Enable Indicates the status of layer 4 destination port matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4DP0] for matching. 0b - Disabled 1b - Enabled
19	Layer 4 Source Port Inverse Match Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
L4SPIM0	Indicates the status of layer 4 source port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] is enabled for perfect matching. This field is valid and applicable only when MAC_Layer4_Address0[L4SP0] = 1. 0b - Disabled 1b - Enabled
18 L4SPM0	Layer 4 Source Port Match Enable Indicates the status of layer 4 source port matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4SP0] for matching. 0b - Disabled 1b - Enabled
17 —	Reserved
16 L4PEN0	Layer 4 Protocol Enable Indicates the status of layer 4 protocol. • If this field is 1, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] are used for matching UDP packets. • If this field is 0, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] are used for matching TCP packets. Layer 4 matching is performed only when the L4SPM0 field or the L4DPM0 field is 1. 0b - Disabled 1b - Enabled
15-11 L3HDBM0	Layer 3 IP DA Higher Bits Match IPv4 packets: This field contains the number of higher bits of IP DA that are matched in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • - .. • 31: All bits except MSb are masked. IPv6 packets:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Bits [12:11] of this field correspond to bits [6:5] of the L3HSBM0 field, which indicates the number of lower bits of IP SA or DA that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 fields: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 field or the L3SAM0 field is 1.
10-6 L3HSBM0	Layer 3 IP SA Higher Bits Match IPv4 packets: This field contains the number of lower bits of IP SA that are masked for matching in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 packets: This field contains bits [4:0] of L3HSBM0. These bits indicate the number of higher bits of IP SA or DA matched in the IPv6 packets. The field is valid and applicable only when the L3DAM0 field or the L3SAM0 field is 1.
5 L3DAIM0	Layer 3 IP DA Inverse Match Enable Indicates the status of layer 3 IP DA inverse matching. • If this field is 1, layer 3 IP DA is enabled for inverse matching. • If this field is 0, layer 3 IP DA is enabled for perfect matching. This field is valid and applicable only if L3DAM0 = 1. 0b - Disabled 1b - Enabled
4 L3DAM0	Layer 3 IP DA Match Enable Indicates the status of layer 3 IP DA matching. • If this field is 1, layer 3 IP DA is enabled for matching. • If this field is 0, MAC ignores layer 3 IP DA for matching.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If L3PEN0 = 1, you must write 1 to either this field or the L3SAM0 field because either the IPv6 DA or SA can be checked for filtering. 0b - Disabled 1b - Enabled
3 L3SAIM0	Layer 3 IP SA Inverse Match Enable Indicates the status of layer 3 IP SA inverse matching. • If this field is 1, layer 3 IP SA is enabled for inverse matching. • If this field is 0, layer 3 IP SA is enabled for perfect matching. This field is valid and applicable only if the L3SAM0 field is 1. 0b - Disabled 1b - Enabled
2 L3SAM0	Layer 3 IP SA Match Enable Indicates the status of layer 3 IP SA matching. • If this field is 1, layer 3 IP SA matching is enabled. • If this field is 0, MAC ignores layer 3 IP SA matching. **NOTE** If L3PEN0 = 1, you must write 1 to either this field or the L3DAM0 field because either IPv6 SA or DA can be checked for filtering. 0b - Disabled 1b - Enabled
1 —	Reserved
0 L3PEN0	Layer 3 Protocol Enable Indicates the status of layer 3 protocol. • If this field is 1, the layer 3 IP SA or DA matching is enabled for IPv6 packets. • If this field is 0, the layer 3 IP SA or DA matching is enabled for IPv4 packets. The layer 3 matching is performed only when either the L3SAM0 field or the L3DAM0 field is 1. 0b - Disabled 1b - Enabled

72.18.125 MAC Layer 4 Address 0 (MAC_Layer4_Address0)

Offset

Register	Offset
MAC_Layer4_Address0	904h

Function

Provides the layer 4 source and destination port numbers.

MAC Layer 4 Address 1 (MAC_Layer4_Address1), MAC L3 L4 Control 1 (MAC_L3_L4_Control1), MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1), MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1), MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1), and MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1) are reserved registers (RO with a default value) if MAC_Packet_Filter[IPFE] is 0 when configuring the core.

You can configure the layer 3 and layer 4 address registers to be double-synchronized by selecting the synchronize layer 3 and layer 4 address registers to the receive clock domain option while configuring the core. When you select this option, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the layer 3 and layer 4 address registers are written to. For proper synchronization updates, you must perform consecutive writes to the same layer 3 and layer 4 address registers after a delay of at least four destination clock cycles.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	L4DP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L4SP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 L4DP0	Layer 4 Destination Port Number Indicates layer 4 destination port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4DPM0] = 1, this field contains the value to be matched with the TCP destination port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4DPM0] are 1, this field contains the value to be matched with the UDP destination port number field in the IPv4 or IPv6 packets.
15-0 L4SP0	Layer 4 Source Port Number Indicates layer 4 source port number.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4SPM0] = 1, this field contains the value to be matched with the TCP source port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4SPM0] are 1, this field contains the value to be matched with the UDP source port number field in the IPv4 or IPv6 packets.

72.18.126 MAC Layer 3 Address 0 Reg 0 (MAC_Layer3_Addr0_Reg0)

Offset

Register	Offset
MAC_Layer3_Addr0_Reg0	910h

Function

Contains the 32-bit IP source address field for IPv4 packets. For IPv6 packets, the field contains bits [31:0] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A00															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A00															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A00	Layer 3 Address 0 Indicates layer 3 address 0. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

72.18.127 MAC Layer 3 Address 1 Reg 0 (MAC_Layer3_Addr1_Reg0)

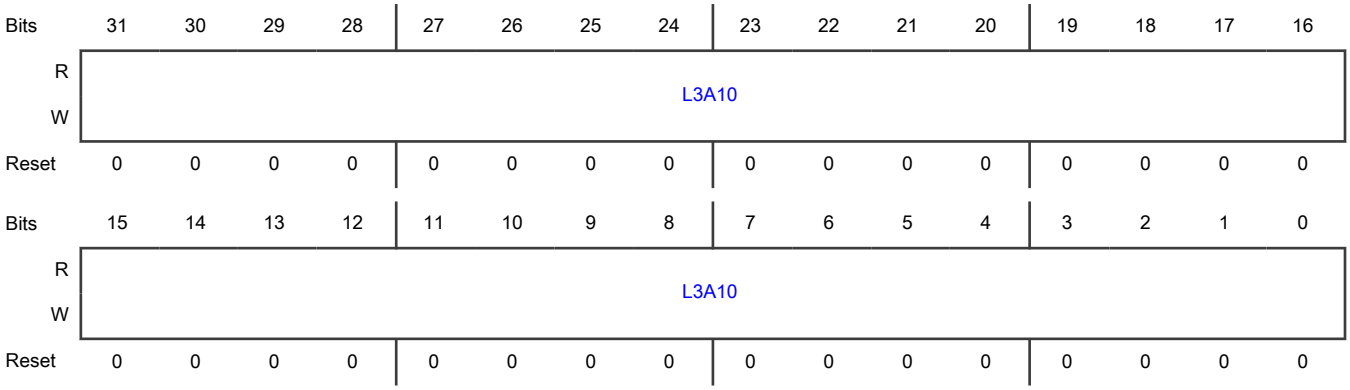
Offset

Register	Offset
MAC_Layer3_Addr1_Reg0	914h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [63:32] of the 128-bit IP source address or destination address field.

Diagram



Fields

Field	Function
31-0 L3A10	Layer 3 Address 1 Indicates layer 3 address 1. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

72.18.128 MAC Layer 3 Address 2 Reg 0 (MAC_Layer3_Addr2_Reg0)

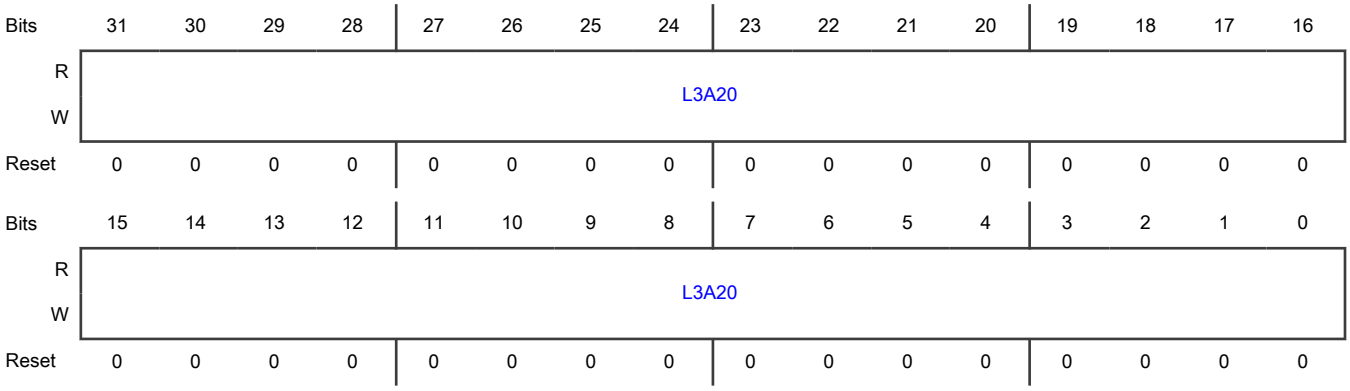
Offset

Register	Offset
MAC_Layer3_Addr2_Reg0	918h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [95:64] of the 128-bit IP source address or destination address field.

Diagram



Fields

Field	Function
31-0 L3A20	Layer 3 Address 2 Indicates layer 3 address 2. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

72.18.129 MAC Layer 3 Address 3 Reg 0 (MAC_Layer3_Addr3_Reg0)

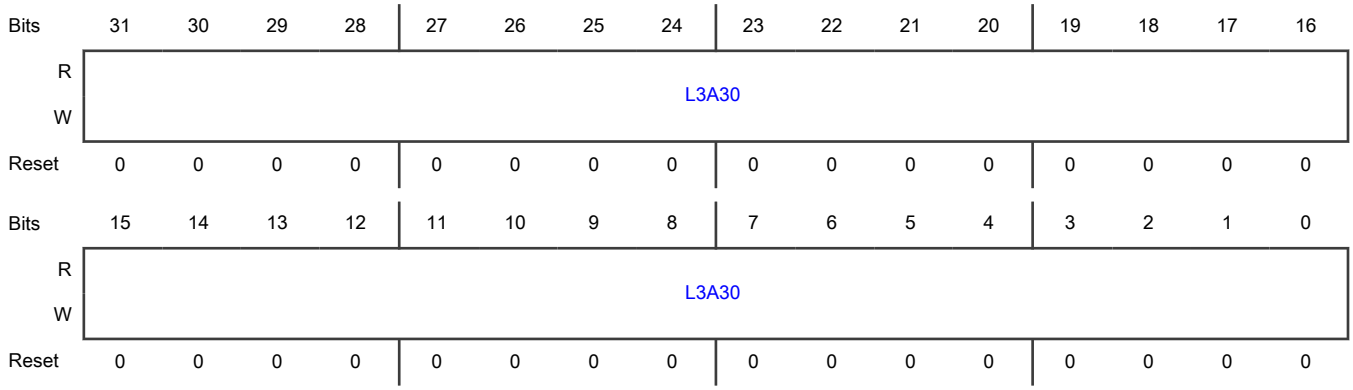
Offset

Register	Offset
MAC_Layer3_Addr3_Reg0	91Ch

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [127:96] of the 128-bit IP source address or destination address field.

Diagram



Fields

Field	Function
31-0 L3A30	Layer 3 Address 3 Indicates layer 3 address 3. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

72.18.130 MAC L3 L4 Control 1 (MAC_L3_L4_Control1)

Offset

Register	Offset
MAC_L3_L4_Control1	930h

Function

Controls the filter 0 operations of the layer 3 and layer 4 protocols.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			DMCHEN1	Reserved			DMCHN1	Reserved		L4DPIM1	L4DPM1	L4SPIM1	L4SPM1	Reserved	L4PEN1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3HDBM1				L3HSBM1						L3DAIM1	L3DAM1	L3SAIM1	L3SAM1	Reserved	L3PEN1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN1	DMA Channel Select Enable 1 Indicates the status of the DMA channel number. • If this field is 1, it enables the selection of the DMA channel number for the packet that the L3_L4 filter passes. The DMCHN fields indicate the DMA channel number. • If this field is 0, the filter does not decide the DMA channel number. 0b - Disabled 1b - Enabled
27-25 —	Reserved
24 DMCHN1	DMA Channel Number 1 Indicates the DMA channel number. If the value of the DMCHEN fields is 1, this field selects the DMA channel number to which the packet that this filter passes is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM1	Layer 4 Destination Port Inverse Match Enable 1 Indicates the status of layer 4 destination port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4DP0] is enabled for perfect matching. This field is valid and applicable only when the L4DPM0 field is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
20 L4DPM1	Layer 4 Destination Port Match Enable 1 Indicates the status of layer 4 destination port matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4DP0] for matching. 0b - Disabled 1b - Enabled
19 L4SPIM1	Layer 4 Source Port Inverse Match Enable 1 Indicates the status of layer 4 source port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] is enabled for perfect matching. This field is valid and applicable only when MAC_Layer4_Address0[L4SP0] = 1. 0b - Disabled 1b - Enabled
18 L4SPM1	Layer 4 Source Port Match Enable 1 Indicates the status of layer 4 source port matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4SP0] for matching. 0b - Disabled 1b - Enabled
17 —	Reserved
16 L4PEN1	Layer 4 Protocol Enable 1 Indicates the status of layer 4 protocol. • If this field is 1, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] of UDP packets are used for matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] of TCP packets are used for matching. Layer 4 matching is performed only when the L4SPM0 field or the L4DPM0 field is 1. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-11 L3HDBM1	Layer 3 IP DA Higher Bits Match 1 IPv4 Packets: This field contains the number of higher bits of IP DA that are matched in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 Packets: Bits [12:11] of this field correspond to bits [6:5] of MAC_L3_L4_Control0[L3HSBM0], which indicates the number of lower bits of IP SA or DA that are masked in the IPv6 packets. The following list describes the concatenated values of MAC_L3_L4_Control0[L3HDBM0][1:0] and MAC_L3_L4_Control0[L3HSBM0]: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked • .. • 127: All bits except MSb are masked. This field is valid and applicable only when MAC_L3_L4_Control0[L3DAM0] or MAC_L3_L4_Control0[L3SAM0] is 1.
10-6 L3HSBM1	Layer 3 IP SA Higher Bits Match 1 IPv4 packets: This field contains the number of lower bits of IP SA that are masked for matching in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 packets: This field contains bits [4:0] of MAC_L3_L4_Control0[L3HSBM0]. These bits indicate the number of higher bits of IP SA or DA matched in the IPv6 packets. The field is valid and applicable only when MAC_L3_L4_Control0[L3DAM0] or MAC_L3_L4_Control0[L3SAM0] is 1.
5 L3DAIM1	Layer 3 IP DA Inverse Match Enable 1 Indicates the status of layer 3 IP DA inverse matching.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, layer 3 IP DA is enabled for inverse matching. • If this field is 0, layer 3 IP DA is enabled for perfect matching. This field is valid and applicable only if MAC_L3_L4_Control0[L3DAM0] = 1. 0b - Disabled 1b - Enabled
4 L3DAM1	Layer 3 IP DA Match Enable 1 Indicates the status of layer 3 IP DA matching. • If this field is 1, layer 3 IP DA is enabled for matching. • If this field is 0, MAC ignores layer 3 IP DA for matching. **NOTE** If MAC_L3_L4_Control0[L3PEN0] = 1, you must write 1 to either this field or to MAC_L3_L4_Control0[L3SAM0] because either the IPv6 DA or SA can be checked for filtering. 0b - Disabled 1b - Enabled
3 L3SAIM1	Layer 3 IP SA Inverse Match Enable 1 Indicates the status of layer 3 IP SA inverse matching. • If this field is 1, layer 3 IP SA is enabled for inverse matching. • If this field is 0, layer 3 IP SA is enabled for perfect matching. This field is valid and applicable only if MAC_L3_L4_Control0[L3SAM0] is 1. 0b - Disabled 1b - Enabled
2 L3SAM1	Layer 3 IP SA Match Enable 1 Indicates the status of layer 3 IP SA matching. • If this field is 1, layer 3 IP SA matching is enabled. • If this field is 0, MAC ignores layer 3 IP SA matching. **NOTE** If MAC_L3_L4_Control0[L3PEN0] = 1, you must write 1 to either this field or to MAC_L3_L4_Control0[L3DAM0] because either IPv6 SA or DA can be checked for filtering. 0b - Disabled 1b - Enabled
1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 L3PEN1	Layer 3 Protocol Enable 1 Indicates the status of layer 3 protocol. • If this field is 1, the layer 3 IP SA or DA matching is enabled for IPv6 packets. • If this field is 0, the layer 3 IP SA or DA matching is enabled for IPv4 packets. The layer 3 matching is performed only when either MAC_L3_L4_Control0[L3SAM0] or MAC_L3_L4_Control0[L3DAM0] is 1. 0b - Disabled 1b - Enabled

72.18.131 MAC Layer 4 Address 1 (MAC_Layer4_Address1)

Offset

Register	Offset
MAC_Layer4_Address1	934h

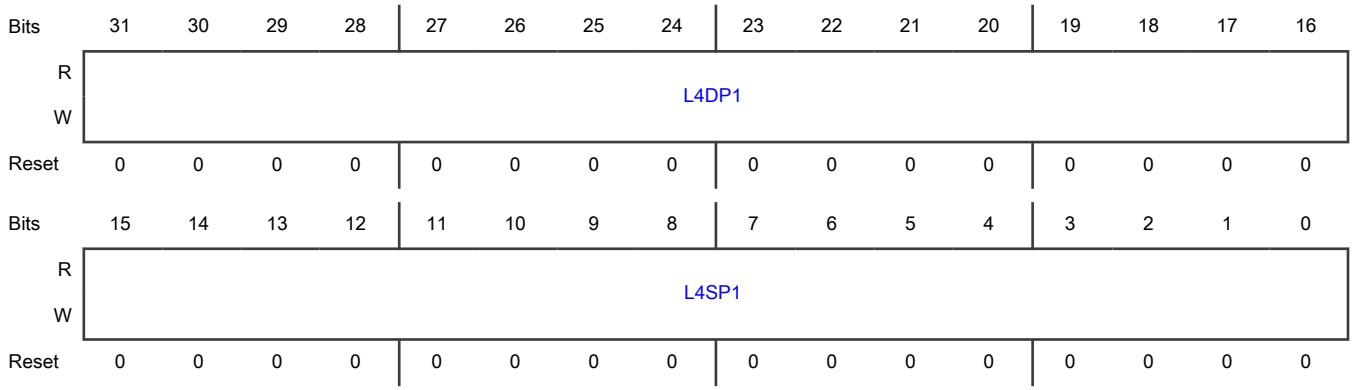
Function

Provides the layer 4 source and destination port numbers.

MAC Layer 4 Address 1 (MAC_Layer4_Address1), MAC L3 L4 Control 1 (MAC_L3_L4_Control1), MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1), MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1), MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1), and MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1) are reserved registers (RO with a default value) if the enable layer 3 and layer 4 packet filter option is not selected when configuring the core.

You can configure the layer 3 and layer 4 address registers to be double-synchronized by selecting the synchronize layer 3 and layer 4 address registers to the receive clock domain option while configuring the core. When you select this option, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the layer 3 and layer 4 address registers are written to. For proper synchronization updates, you must perform consecutive writes to the same layer 3 and layer 4 address registers after a delay of at least four destination clock cycles.

Diagram



Fields

Field	Function
31-16 L4DP1	Layer 4 Destination Port Number 1 Indicates layer 4 destination port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4DPM0] = 1, this field contains the value to be matched with the TCP destination port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4DPM0] are 1, this field contains the value to be matched with the UDP destination port number field in the IPv4 or IPv6 packets.
15-0 L4SP1	Layer 4 Source Port Number 1 Indicates layer 4 source port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4SPM0] = 1, this field contains the value to be matched with the TCP source port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4SPM0] are 1, this field contains the value to be matched with the UDP source port number field in the IPv4 or IPv6 packets.

72.18.132 MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1)

Offset

Register	Offset
MAC_Layer3_Addr0_Reg1	940h

Function

Contains the 32-bit IP source address field for IPv4 packets. For IPv6 packets, the field contains bits [31:0] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A01															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A01															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A01	Layer 3 Address 0 Indicates layer 3 address 0. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

72.18.133 MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1)

Offset

Register	Offset
MAC_Layer3_Addr1_Reg1	944h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits[63:32] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A11															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A11															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A11	Layer 3 Address 1 Indicates layer 3 address 1.

Table continues on the next page...

Field	Function
	• If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

72.18.134 MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1)

Offset

Register	Offset
MAC_Layer3_Addr2_Reg1	948h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [95:64] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A21															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A21															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A21	Layer 3 Address 2 Indicates layer 3 address 2. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

72.18.135 MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1)

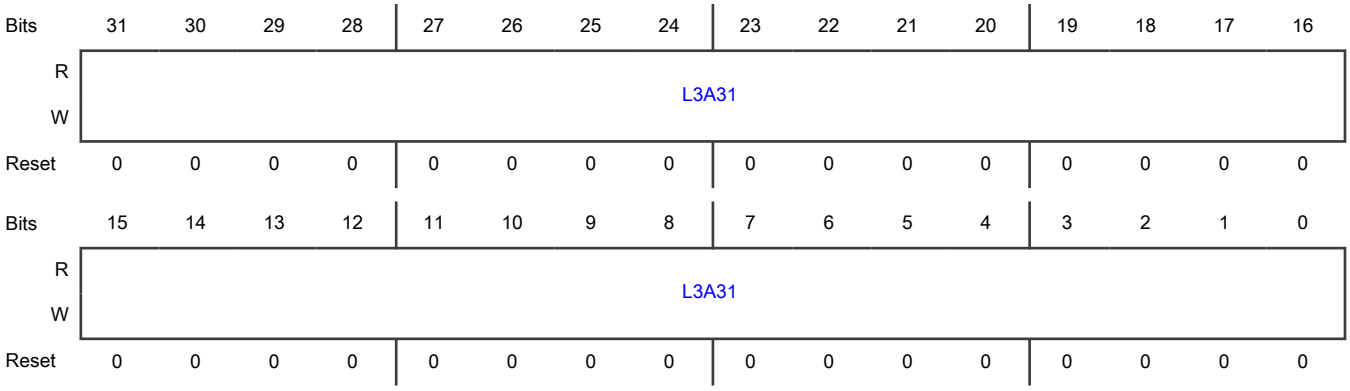
Offset

Register	Offset
MAC_Layer3_Addr3_Reg1	94Ch

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [127:96] of the 128-bit IP source address or destination address field.

Diagram



Fields

Field	Function
31-0 L3A31	Layer 3 Address 3 Indicates layer 3 address 3. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

72.18.136 MAC L3 L4 Control 2 (MAC_L3_L4_Control2)

Offset

Register	Offset
MAC_L3_L4_Control2	960h

Function

Controls the filter 0 operations of the layer 3 and layer 4 protocols.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			DMCHEN2	Reserved			DMCHN2	Reserved		L4DPIM2	L4DPM2	L4SPIM2	L4SPM2	Reserved	L4PEN2
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3HDBM2				L3HSBM2						L3DAIM2	L3DAM2	L3SAIM2	L3SAM2	Reserved	L3PEN2
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN2	DMA Channel Select Enable 2 Indicates the status of the DMA channel number. • If this field is 1, it enables the selection of the DMA channel number for the packet that the L3_L4 filter passes. The DMCHN fields indicate the DMA channel number. • If this field is 0, the filter does not decide the DMA channel number. 0b - Disabled 1b - Enabled
27-25 —	Reserved
24 DMCHN2	DMA Channel Number 2 Indicates the DMA channel number. If the value of the DMCHEN fields is 1, this field selects the DMA channel number to which the packet that this filter passes is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM2	Layer 4 Destination Port Inverse Match Enable 2 Indicates the status of layer 4 destination port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4DP0] is enabled for perfect matching. This field is valid and applicable only when the L4DPM0 field is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
20 L4DPM2	Layer 4 Destination Port Match Enable 2 Indicates the status of layer 4 destination port matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4DP0] for matching. 0b - Disabled 1b - Enabled
19 L4SPIM2	Layer 4 Source Port Inverse Match Enable 2 Indicates the status of layer 4 source port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] is enabled for perfect matching. This field is valid and applicable only when MAC_Layer4_Address0[L4SP0] = 1. 0b - Disabled 1b - Enabled
18 L4SPM2	Layer 4 Source Port Match Enable 2 Indicates the status of layer 4 source port matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4SP0] for matching. 0b - Disabled 1b - Enabled
17 —	Reserved
16 L4PEN2	Layer 4 Protocol Enable 2 Indicates the status of layer 4 protocol. • If this field is 1, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] of UDP packets are used for matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] of TCP packets are used for matching. Layer 4 matching is performed only when the L4SPM0 field or the L4DPM0 field is 1. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-11 L3HDBM2	Layer 3 IP DA Higher Bits Match 2 IPv4 Packets: This field contains the number of higher bits of IP DA that are matched in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 Packets: Bits [12:11] of this field correspond to bits [6:5] of MAC_L3_L4_Control0[L3HSBM0], which indicates the number of lower bits of IP SA or DA that are masked in the IPv6 packets. The following list describes the concatenated values of MAC_L3_L4_Control0[L3HDBM0][1:0] and MAC_L3_L4_Control0[L3HSBM0]: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked • .. • 127: All bits except MSb are masked. This field is valid and applicable only when MAC_L3_L4_Control0[L3DAM0] or MAC_L3_L4_Control0[L3SAM0] is 1.
10-6 L3HSBM2	Layer 3 IP SA Higher Bits Match 2 IPv4 packets: This field contains the number of lower bits of IP SA that are masked for matching in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 packets: This field contains bits [4:0] of MAC_L3_L4_Control0[L3HSBM0]. These bits indicate the number of higher bits of IP SA or DA matched in the IPv6 packets. The field is valid and applicable only when MAC_L3_L4_Control0[L3DAM0] or MAC_L3_L4_Control0[L3SAM0] is 1.
5 L3DAIM2	Layer 3 IP DA Inverse Match Enable 2 Indicates the status of layer 3 IP DA inverse matching.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, layer 3 IP DA is enabled for inverse matching. • If this field is 0, layer 3 IP DA is enabled for perfect matching. This field is valid and applicable only if MAC_L3_L4_Control0[L3DAM0] = 1. 0b - Disabled 1b - Enabled
4 L3DAM2	Layer 3 IP DA Match Enable 2 Indicates the status of layer 3 IP DA matching. • If this field is 1, layer 3 IP DA is enabled for matching. • If this field is 0, MAC ignores layer 3 IP DA for matching. **NOTE** If MAC_L3_L4_Control0[L3PEN0] = 1, you must write 1 to either this field or to MAC_L3_L4_Control0[L3SAM0] because either the IPv6 DA or SA can be checked for filtering. 0b - Disabled 1b - Enabled
3 L3SAIM2	Layer 3 IP SA Inverse Match Enable 2 Indicates the status of layer 3 IP SA inverse matching. • If this field is 1, layer 3 IP SA is enabled for inverse matching. • If this field is 0, layer 3 IP SA is enabled for perfect matching. This field is valid and applicable only if MAC_L3_L4_Control0[L3SAM0] is 1. 0b - Disabled 1b - Enabled
2 L3SAM2	Layer 3 IP SA Match Enable 2 Indicates the status of layer 3 IP SA matching. • If this field is 1, layer 3 IP SA matching is enabled. • If this field is 0, MAC ignores layer 3 IP SA matching. **NOTE** If MAC_L3_L4_Control0[L3PEN0] = 1, you must write 1 to either this field or to MAC_L3_L4_Control0[L3DAM0] because either IPv6 SA or DA can be checked for filtering. 0b - Disabled 1b - Enabled
1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 L3PEN2	Layer 3 Protocol Enable 2 Indicates the status of layer 3 protocol. • If this field is 1, the layer 3 IP SA or DA matching is enabled for IPv6 packets. • If this field is 0, the layer 3 IP SA or DA matching is enabled for IPv4 packets. The layer 3 matching is performed only when either MAC_L3_L4_Control0[L3SAM0] or MAC_L3_L4_Control0[L3DAM0] is 1. 0b - Disabled 1b - Enabled

72.18.137 MAC Layer 4 Address 2 (MAC_Layer4_Address2)

Offset

Register	Offset
MAC_Layer4_Address2	964h

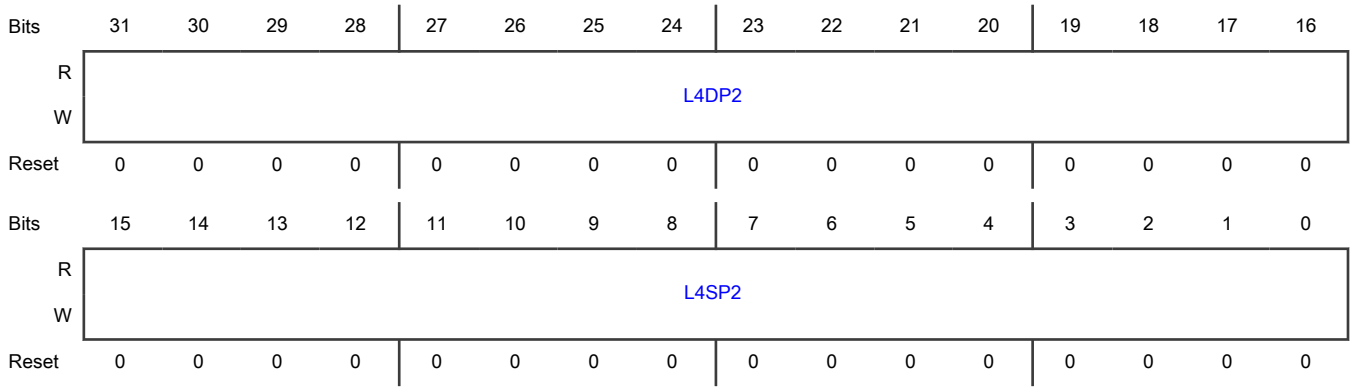
Function

Provides the layer 4 source and destination port numbers.

MAC Layer 4 Address 1 (MAC_Layer4_Address1), MAC L3 L4 Control 1 (MAC_L3_L4_Control1), MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1), MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1), MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1), and MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1) are reserved registers (RO with a default value) if the enable layer 3 and layer 4 packet filter option is not selected when configuring the core.

You can configure the layer 3 and layer 4 address registers to be double-synchronized by selecting the synchronize layer 3 and layer 4 address registers to the receive clock domain option while configuring the core. When you select this option, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the layer 3 and layer 4 address registers are written to. For proper synchronization updates, you must perform consecutive writes to the same layer 3 and layer 4 address registers after a delay of at least four destination clock cycles.

Diagram



Fields

Field	Function
31-16 L4DP2	Layer 4 Destination Port Number 2 Indicates layer 4 destination port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4DPM0] = 1, this field contains the value to be matched with the TCP destination port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4DPM0] are 1, this field contains the value to be matched with the UDP destination port number field in the IPv4 or IPv6 packets.
15-0 L4SP2	Layer 4 Source Port Number 2 Indicates layer 4 source port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4SPM0] = 1, this field contains the value to be matched with the TCP source port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4SPM0] are 1, this field contains the value to be matched with the UDP source port number field in the IPv4 or IPv6 packets.

72.18.138 MAC Layer 3 Address 0 Reg 2 (MAC_Layer3_Addr0_Reg2)

Offset

Register	Offset
MAC_Layer3_Addr0_Reg2	970h

Function

Contains the 32-bit IP source address field for IPv4 packets. For IPv6 packets, the field contains bits [31:0] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A02															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A02															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A02	Layer 3 Address 0 Indicates layer 3 address 0. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

72.18.139 MAC Layer 3 Address 1 Reg 2 (MAC_Layer3_Addr1_Reg2)

Offset

Register	Offset
MAC_Layer3_Addr1_Reg2	974h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits[63:32] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A12															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A12															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A12	Layer 3 Address 1 Indicates layer 3 address 1.

Table continues on the next page...

Field	Function
	• If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

72.18.140 MAC Layer 3 Address 2 Reg 2 (MAC_Layer3_Addr2_Reg2)

Offset

Register	Offset
MAC_Layer3_Addr2_Reg2	978h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [95:64] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A22															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A22															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A22	Layer 3 Address 2 Indicates layer 3 address 2. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

72.18.141 MAC Layer 3 Address 3 Reg 2 (MAC_Layer3_Addr3_Reg2)

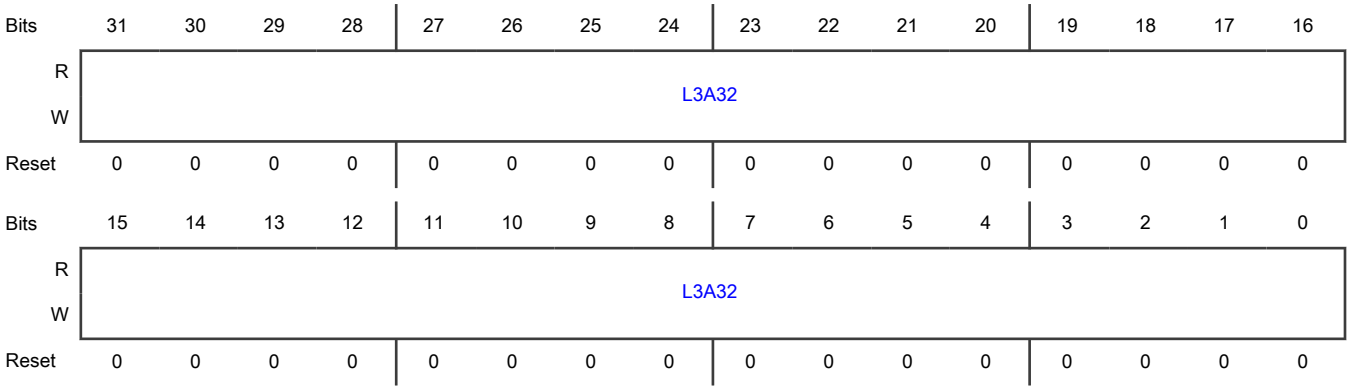
Offset

Register	Offset
MAC_Layer3_Addr3_Reg2	97Ch

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [127:96] of the 128-bit IP source address or destination address field.

Diagram



Fields

Field	Function
31-0 L3A32	Layer 3 Address 3 Indicates layer 3 address 3. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

72.18.142 MAC L3 L4 Control 3 (MAC_L3_L4_Control3)

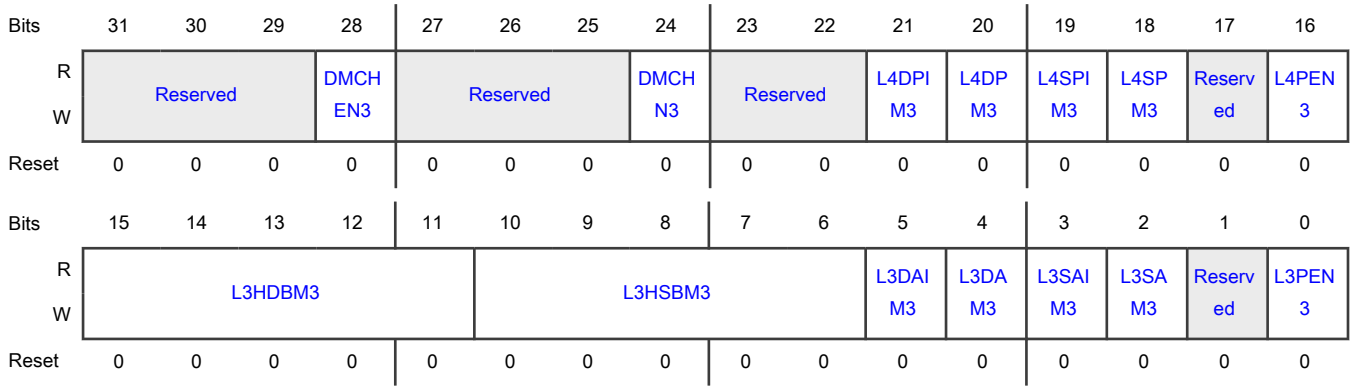
Offset

Register	Offset
MAC_L3_L4_Control3	990h

Function

Controls the filter 0 operations of the layer 3 and layer 4 protocols.

Diagram



Fields

Field	Function
31-29 —	Reserved
28 DMCHEN3	DMA Channel Select Enable 3 Indicates the status of the DMA channel number. • If this field is 1, it enables the selection of the DMA channel number for the packet that the L3_L4 filter passes. The DMCHN fields indicate the DMA channel number. • If this field is 0, the filter does not decide the DMA channel number. 0b - Disabled 1b - Enabled
27-25 —	Reserved
24 DMCHN3	DMA Channel Number 2 Indicates the DMA channel number. If the value of the DMCHEN fields is 1, this field selects the DMA channel number to which the packet that this filter passes is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM3	Layer 4 Destination Port Inverse Match Enable 3 Indicates the status of layer 4 destination port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4DP0] is enabled for perfect matching. This field is valid and applicable only when the L4DPM0 field is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
20 L4DPM3	Layer 4 Destination Port Match Enable 3 Indicates the status of layer 4 destination port matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4DP0] for matching. 0b - Disabled 1b - Enabled
19 L4SPIM3	Layer 4 Source Port Inverse Match Enable 3 Indicates the status of layer 4 source port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] is enabled for perfect matching. This field is valid and applicable only when MAC_Layer4_Address0[L4SP0] = 1. 0b - Disabled 1b - Enabled
18 L4SPM3	Layer 4 Source Port Match Enable 3 Indicates the status of layer 4 source port matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4SP0] for matching. 0b - Disabled 1b - Enabled
17 —	Reserved
16 L4PEN3	Layer 4 Protocol Enable 3 Indicates the status of layer 4 protocol. • If this field is 1, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] of UDP packets are used for matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] of TCP packets are used for matching. Layer 4 matching is performed only when the L4SPM0 field or the L4DPM0 field is 1. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-11 L3HDBM3	Layer 3 IP DA Higher Bits Match 3 IPv4 Packets: This field contains the number of higher bits of IP DA that are matched in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 Packets: Bits [12:11] of this field correspond to bits [6:5] of MAC_L3_L4_Control0[L3HSBM0], which indicates the number of lower bits of IP SA or DA that are masked in the IPv6 packets. The following list describes the concatenated values of MAC_L3_L4_Control0[L3HDBM0][1:0] and MAC_L3_L4_Control0[L3HSBM0]: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked • .. • 127: All bits except MSb are masked. This field is valid and applicable only when MAC_L3_L4_Control0[L3DAM0] or MAC_L3_L4_Control0[L3SAM0] is 1. .
10-6 L3HSBM3	Layer 3 IP SA Higher Bits Match 3 IPv4 packets: This field contains the number of lower bits of IP SA that are masked for matching in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 packets: This field contains bits [4:0] of MAC_L3_L4_Control0[L3HSBM0]. These bits indicate the number of higher bits of IP SA or DA matched in the IPv6 packets. The field is valid and applicable only when MAC_L3_L4_Control0[L3DAM0] or MAC_L3_L4_Control0[L3SAM0] is 1.
5	Layer 3 IP DA Inverse Match Enable 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
L3DAIM3	Indicates the status of layer 3 IP DA inverse matching. • If this field is 1, layer 3 IP DA is enabled for inverse matching. • If this field is 0, layer 3 IP DA is enabled for perfect matching. This field is valid and applicable only if MAC_L3_L4_Control0[L3DAM0] = 1. 0b - Disabled 1b - Enabled
4 L3DAM3	Layer 3 IP DA Match Enable 3 Indicates the status of layer 3 IP DA matching. • If this field is 1, layer 3 IP DA is enabled for matching. • If this field is 0, MAC ignores layer 3 IP DA for matching. **NOTE** If MAC_L3_L4_Control0[L3PEN0] = 1, you must write 1 to either this field or to MAC_L3_L4_Control0[L3SAM0] because either the IPv6 DA or SA can be checked for filtering. 0b - Disabled 1b - Enabled
3 L3SAIM3	Layer 3 IP SA Inverse Match Enable 3 Indicates the status of layer 3 IP SA inverse matching. • If this field is 1, layer 3 IP SA is enabled for inverse matching. • If this field is 0, layer 3 IP SA is enabled for perfect matching. This field is valid and applicable only if MAC_L3_L4_Control0[L3SAM0] is 1. 0b - Disabled 1b - Enabled
2 L3SAM3	Layer 3 IP SA Match Enable 3 Indicates the status of layer 3 IP SA matching. • If this field is 1, layer 3 IP SA matching is enabled. • If this field is 0, MAC ignores layer 3 IP SA matching. **NOTE** If MAC_L3_L4_Control0[L3PEN0] = 1, you must write 1 to either this field or to MAC_L3_L4_Control0[L3DAM0] because either IPv6 SA or DA can be checked for filtering. 0b - Disabled 1b - Enabled
1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 L3PEN3	Layer 3 Protocol Enable 3 Indicates the status of layer 3 protocol. • If this field is 1, the layer 3 IP SA or DA matching is enabled for IPv6 packets. • If this field is 0, the layer 3 IP SA or DA matching is enabled for IPv4 packets. The layer 3 matching is performed only when either MAC_L3_L4_Control0[L3SAM0] or MAC_L3_L4_Control0[L3DAM0] is 1. 0b - Disabled 1b - Enabled

72.18.143 MAC Layer 4 Address 3 (MAC_Layer4_Address3)

Offset

Register	Offset
MAC_Layer4_Address3	994h

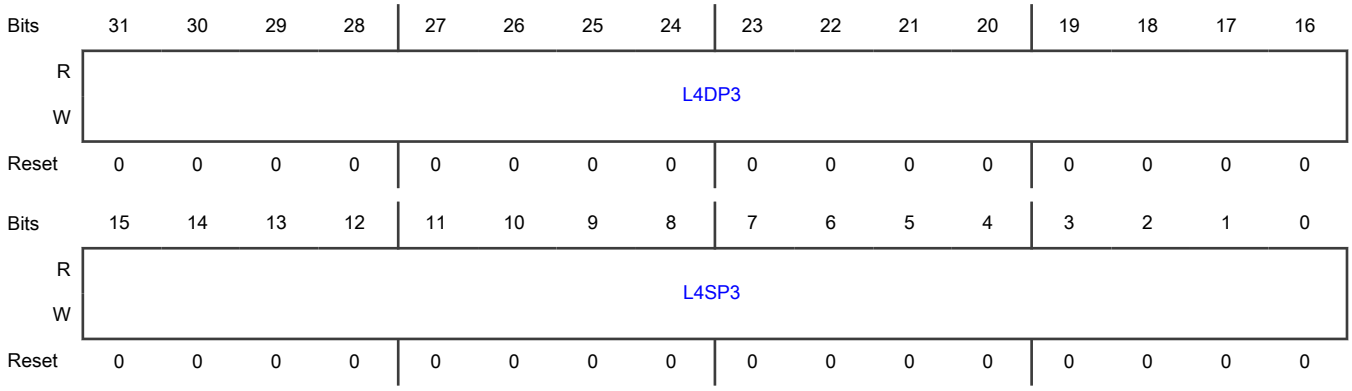
Function

Provides the layer 4 source and destination port numbers.

MAC Layer 4 Address 1 (MAC_Layer4_Address1), MAC L3 L4 Control 1 (MAC_L3_L4_Control1), MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1), MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1), MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1), and MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1) are reserved registers (RO with a default value) if the enable layer 3 and layer 4 packet filter option is not selected when configuring the core.

You can configure the layer 3 and layer 4 address registers to be double-synchronized by selecting the synchronize layer 3 and layer 4 address registers to the receive clock domain option while configuring the core. When you select this option, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the layer 3 and layer 4 address registers are written to. For proper synchronization updates, you must perform consecutive writes to the same layer 3 and layer 4 address registers after a delay of at least four destination clock cycles.

Diagram



Fields

Field	Function
31-16 L4DP3	Layer 4 Destination Port Number 3 Indicates layer 4 destination port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4DPM0] = 1, this field contains the value to be matched with the TCP destination port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4DPM0] are 1, this field contains the value to be matched with the UDP destination port number field in the IPv4 or IPv6 packets.
15-0 L4SP3	Layer 4 Source Port Number 3 Indicates layer 4 source port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4SPM0] = 1, this field contains the value to be matched with the TCP source port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4SPM0] are 1, this field contains the value to be matched with the UDP source port number field in the IPv4 or IPv6 packets.

72.18.144 MAC Layer 3 Address 0 Reg 3 (MAC_Layer3_Addr0_Reg3)

Offset

Register	Offset
MAC_Layer3_Addr0_Reg3	9A0h

Function

Contains the 32-bit IP source address field for IPv4 packets. For IPv6 packets, the field contains bits [31:0] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A03															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A03															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A03	Layer 3 Address 0 Indicates layer 3 address 0. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

72.18.145 MAC Layer 3 Address 1 Reg 3 (MAC_Layer3_Addr1_Reg3)

Offset

Register	Offset
MAC_Layer3_Addr1_Reg3	9A4h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [63:32] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A13															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A13															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A13	Layer 3 Address 1 Indicates layer 3 address 1.

Table continues on the next page...

Field	Function
	• If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

72.18.146 MAC Layer 3 Address 2 Reg 3 (MAC_Layer3_Addr2_Reg3)

Offset

Register	Offset
MAC_Layer3_Addr2_Reg3	9A8h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [95:64] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A23															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A23															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A23	Layer 3 Address 2 Indicates layer 3 address 2. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

72.18.147 MAC Layer 3 Address 3 Reg 3 (MAC_Layer3_Addr3_Reg3)

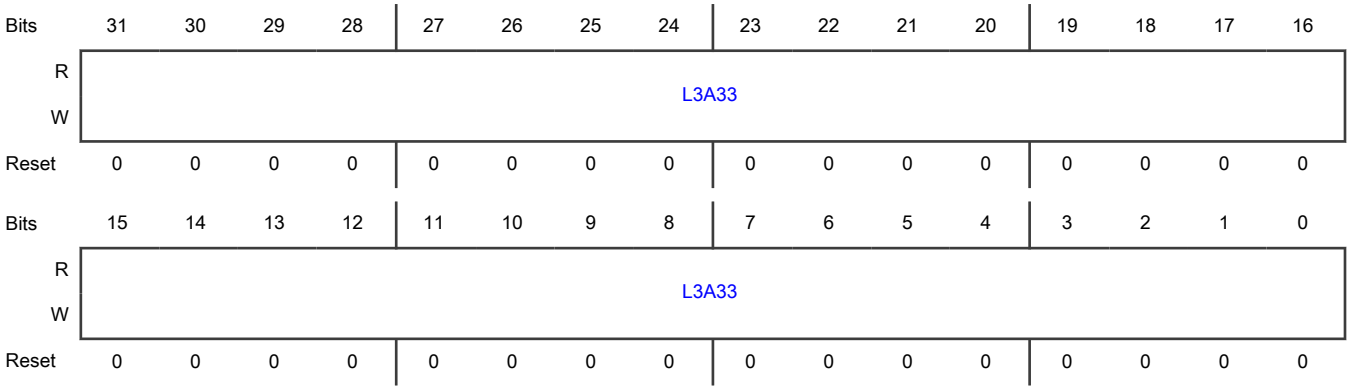
Offset

Register	Offset
MAC_Layer3_Addr3_Reg3	9ACh

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [127:96] of the 128-bit IP source address or destination address field.

Diagram



Fields

Field	Function
31-0 L3A33	Layer 3 Address 3 Indicates layer 3 address 3. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

72.18.148 MAC Timestamp Control (MAC_Timestamp_Control)

Offset

Register	Offset
MAC_Timestamp_Control	B00h

Function

Controls the operation of the system time generator and processing of PTP packets for timestamping in the receiver.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			AV8021A...	Reserved			TXTSSTSM	Reserved			ESTI	Reserved	TSENMAC...	SNAPTYPSEL	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSMSTRE...	TSEVNTE...	TSIPV4E...	TSIPV6E...	TSIPENA	TSVER2E...	TSCTRLS...	TSENALL	Reserved	PTGE	TSADDREG	Reserved	TSUPDT	TSINIT	TSCFUPDT	TSENA
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 AV8021ASMEN	AV 802.1AS Mode Enable Indicates the status of AV 802.1AS mode. If this field is 1, MAC processes only untagged PTP over Ethernet packets for providing PTP status and capturing timestamp snapshots, which means using the IEEE 802.1AS mode of operation. If MAC_HW_Feature1[PTOEN] = 1, for the purpose of PTP offload, the transport-specific field in the PTP header is generated and checked based on the value of the AV8021ASMEN field. 0b - Disabled 1b - Enabled
27-25 —	Reserved
24 TXTSSTSM	Transmit Timestamp Status Mode Indicates the status of transmit timestamp status mode. • If this field is 1, MAC overwrites the previous transmit timestamp status even if the software does not read it. MAC indicates this by writing 1 to MAC_Tx_Timestamp_Status_Nanoseconds[TXTSSMIS]. • If this field is 0, MAC ignores the timestamp status of the current packet if the software does not read the timestamp status of the previous packet. MAC indicates this by writing 1 to MAC_Tx_Timestamp_Status_Nanoseconds[TXTSSMIS]. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-21 —	Reserved
20 ESTI	External System Time Input Indicates the status of external system time input. If this field is 1, MAC uses the external 64-bit reference system time input for these functions: • To consider the timestamp provided as status • To insert the timestamp in transmit PTP packets if the one-step timestamp or the timestamp offload feature is enabled If this field is 1, MAC uses the internal reference system time. 0b - Disabled 1b - Enabled
19 —	Reserved
18 TSENMACADDR	Enable MAC Address For PTP Packet Filtering Indicates whether the MAC address for PTP packet filtering is enabled. If this field is 1, the DA MAC address, which matches any MAC Address register, is used to filter the PTP packets when PTP is directly sent over Ethernet. If this field is 1, received PTP packets with DA containing a special multicast or unicast address that matches the one programmed in the MAC address registers are considered for processing when PTP is directly sent over Ethernet. • For a normal time stamping operation, MAC address registers 0 to 31 are considered for unicast destination address matching. • For PTP offload, only MAC address register 0 is considered for unicast destination address matching. 0b - Disabled 1b - Enabled
17-16 SNAPTYPSEL	Select PTP Packets For Taking Snapshots Determines the set of PTP packet types for which a snapshot needs to be taken. This field, along with the TSEVNTENA and TSMSTRENA fields, determines the set of PTP packet types for which a snapshot needs to be taken. See Receive path functions for encoding.
15 TSMSTRENA	Enable Snapshot For Messages Relevant To Master Indicates whether the snapshot for messages relevant to master is enabled. If this field is 1, the snapshot is taken only for the messages that are relevant to the master node. Otherwise, the snapshot is taken for the messages relevant to the slave node.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
14 TSEVNTENA	Enable Timestamp Snapshot For Event Messages Enables or disables timestamp snapshot for event messages. If this field is 1, the timestamp snapshot is taken only for event messages (SYNC, Delay_Req, Pdelay_Req, or Pdelay_Resp). When the field becomes 0, the snapshot is taken for all the messages except Announce, Management, and Signaling. For more information on timestamp snapshot dependency on register bits, see Receive path functions. 0b - Disabled 1b - Enabled
13 TSIPV4ENA	Enable Processing Of PTP Packets Sent Over IPv4-UDP Indicates whether the processing of PTP packets sent over IPv4-UDP is enabled. If this field is 1, the MAC receiver processes the PTP packets encapsulated in the IPv4-UDP packets. When this field becomes 0, MAC ignores the PTP transported over the IPv4-UDP packets. The default value of this field is 1. 0b - Disabled 1b - Enabled
12 TSIPV6ENA	Enable Processing Of PTP Packets Sent Over IPv6-UDP Indicates whether the processing of PTP packets sent over IPv6-UDP is enabled. If this field is 1, the MAC receiver processes the PTP packets encapsulated in the IPv6-UDP packets. If the value is 0, MAC ignores the PTP transported over IPv6-UDP packets. 0b - Disabled 1b - Enabled
11 TSIPENA	Enable Processing Of PTP Over Ethernet Packets Indicates the status of PTP processing over Ethernet packets. If this field is 1, the MAC receiver processes the PTP packets encapsulated directly in the Ethernet packets. When the field becomes 0, MAC ignores the PTP over Ethernet packets. 0b - Disabled 1b - Enabled
10 TSVER2ENA	Enable PTP Packet Processing For Version 2 Format Indicates the status of PTP packet processing for version 2 format. If this field is 1, the IEEE 1588 version 2 format is used to process the PTP packets. When this field becomes 0, the IEEE 1588 version 1 format is used to process the PTP packets. See PTP processing and control for more information on the IEEE 1588 formats.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
9 TSCTRLSSR	Timestamp Digital Or Binary Rollover Control Indicates the status of timestamp digital or binary rollover control. • If this field is 1, the Timestamp Low register rolls over after the 3B9A_C9FFh value (that is, 1 nanosecond accuracy) and increments the timestamp (high) seconds. • If this field is 0, the rollover value of the Subsecond register is 7FFF_FFFFh. The sub-second increment must be programmed correctly depending on the PTP reference clock frequency and the value of this field. 0b - Disabled 1b - Enabled
8 TSENALL	Enable Timestamp For All Packets Indicates the status of timestamp snapshot for all packets. If this field is 1, the timestamp snapshot is enabled for all packets that MAC receives. 0b - Disabled 1b - Enabled
7 —	Reserved
6 PTGE	Presentation Time Generation Enable Indicates the status of presentation time generation. If this field is 1, the presentation time generation is enabled. 0b - Disabled 1b - Enabled
5 TSADDREG	Update Addend Register Indicates whether MAC Timestamp Addend (MAC_Timestamp_Addend) is updated. If this field is 1, the contents of MAC Timestamp Addend (MAC_Timestamp_Addend) are updated in the PTP block for fine correction. The field becomes 0 after the update completes. The value of the field must also be 0 before you write 1 to it. Access restrictions apply to this field that clears automatically. Writing 0 has no effect. 0b - Not updated 1b - Updated
4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 TSUPDT	Update Timestamp Indicates whether the timestamp is updated. If this field is 1, the system time is updated (added or subtracted) with the value specified in MAC System Time Seconds Update (MAC_System_Time_Seconds_Update) and MAC System Time Nanoseconds Update (MAC_System_Time_Nanoseconds_Update). The value of this field must be 0 before you update it. It resets after the update is complete in hardware. MAC_System_Time_Higher_Word_Seconds[TSHWR], if enabled during core configuration, is not updated. When media clock generation and recovery is configured (DWC_EQOS_FLEXI_PPS_OUT_EN) and enabled, MAC Presentation Time Update (MAC_Presn_Time_Updt) must also be updated before you write 1 to this field. Access restrictions apply to this field that clears automatically. Writing 0 has no effect. 0b - Not updated 1b - Updated
2 TSINIT	Initialize Timestamp Indicates whether the timestamp is initialized. If this field is 1, the system time is initialized (overwritten) with the value specified in MAC System Time Seconds Update (MAC_System_Time_Seconds_Update) and MAC System Time Nanoseconds Update (MAC_System_Time_Nanoseconds_Update). The value of this field must be 0 before you update it, and the field is reset after the initialization is complete. MAC_System_Time_Higher_Word_Seconds[TSHWR], if enabled during core configuration, can only be initialized. When media clock generation and recovery is configured (DWC_EQOS_FLEXI_PPS_OUT_EN) and enabled, MAC Presentation Time Update (MAC_Presn_Time_Updt) must also be updated before you write 1 to this field. Access restrictions apply to this field that clears automatically. Writing 0 has no effect. 0b - Not initialized 1b - Initialized
1 TSCFUPDT	Fine Or Coarse Timestamp Update Indicates the method used to update system timestamp. • If this field is 1, the fine method is used to update system timestamp. • If this field is 0, the coarse method is used to update the system timestamp. 0b - Coarse method 1b - Fine method
0 TSENA	Timestamp Enable Enables or disables timestamp for transmit and receive packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, the timestamp is added for transmit and receive packets. • If this field is 0, timestamp is not added for transmit and receive packets and the timestamp generator is also suspended. You need to initialize the timestamp (system time) after enabling this mode. On the receive side, MAC processes the 1588 packets only if this field is 1. 0b - Disabled 1b - Enabled

72.18.149 MAC Sub Second Increment (MAC_Sub_Second_Increment)

Offset

Register	Offset
MAC_Sub_Second_Increment	B04h

Function

Specifies the value to be added to the internal system time register at every cycle of the clk_ptp_ref_i clock.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								SSINC							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SNSINC								Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 SSINC	Sub-Second Increment Value

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Contains the sub-second incement value, which accumulates every clock cycle (of clk_ptp_i) with the contents of the Subsecond register. For example, if the PTP clock is 50 MHz (period is 20 ns), you must program 20 (14h) when MAC System Time In Nanoseconds (MAC_System_Time_Nanoseconds) has an accuracy of 1 ns (MAC_Timestamp_Control[TSCTRLSSR] = 1). When TSCTRLSSR = 0, the Nanoseconds register has a resolution of ~0.465 ns. In this case, you must program a value of 43 (2Bh), which is derived by 20 ns/0.465.
15-8 SNSINC	Sub-Nanosecond Increment Value Contains the sub-nanosecond increment value, represented in nanoseconds multiplied by 2^8. This value is accumulated in the sub-nanoseconds field of the Subsecond register. For example, if MAC_Timestamp_Control[TSCTRLSSR] = 1, and if the required increment is 5.3 ns, then this field must be 4Ch and the SSINC field must be 05h.
7-0 —	Reserved

72.18.150 MAC System Time In Seconds (MAC_System_Time_Seconds)

Offset

Register	Offset
MAC_System_Time_Seconds	B08h

Function

Indicates, along with MAC System Time In Nanoseconds (MAC_System_Time_Nanoseconds), the current value of the system time that MAC maintains. Although it is updated on a continuous basis, there is some delay from the actual time because of clock domain transfer latencies (from clk_ptp_ref_i to the CSR clock).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TSS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TSS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSS	Timestamp Second Indicates the current value, in seconds, of the system time that MAC maintains.

72.18.151 MAC System Time In Nanoseconds (MAC_System_Time_Nanoseconds)

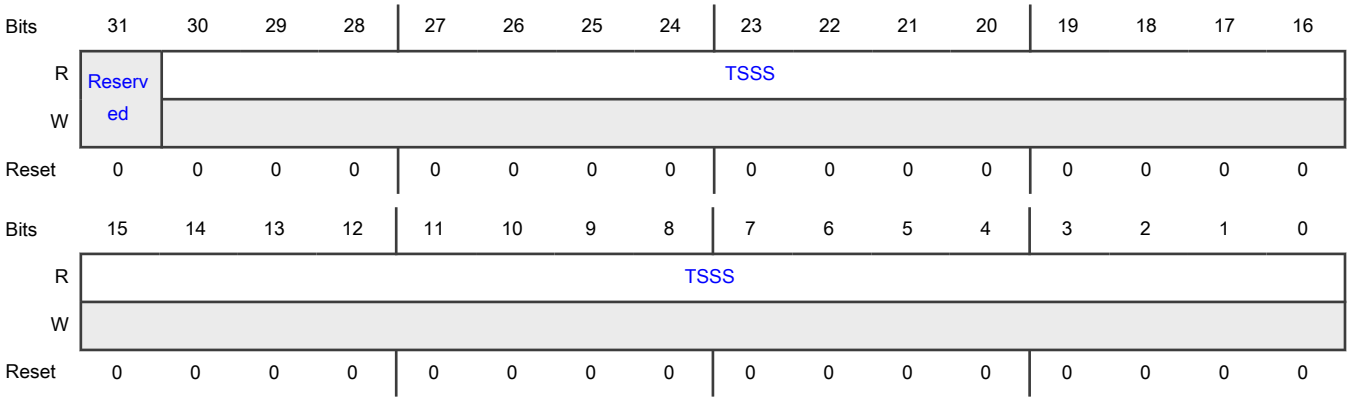
Offset

Register	Offset
MAC_System_Time_Nanoseconds	B0Ch

Function

Indicates, along with MAC System Time In Seconds (MAC_System_Time_Seconds), the current value of the system time that MAC maintains.

Diagram



Fields

Field	Function
31 —	Reserved
30-0 TSSS	Timestamp Sub Seconds Indicates the sub-second representation of time, with an accuracy of 0.46 ns. When MAC_Timestamp_Control[TSCTRLSSR] = 1, each bit represents 1 ns. The maximum value is 3B9A_C9FFh after which it rolls over to 0.

72.18.152 MAC System Time Seconds Update (MAC_System_Time_Seconds_Update)

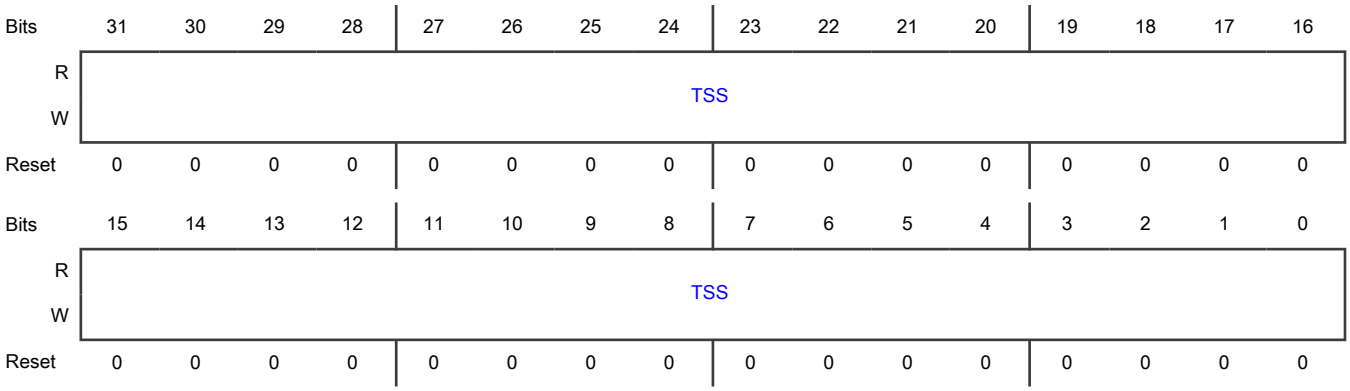
Offset

Register	Offset
MAC_System_Time_Seconds_Update	B10h

Function

Initializes or updates, along with MAC System Time Nanoseconds Update (MAC_System_Time_Nanoseconds_Update), the system time that MAC maintains. You must write both to this register and the MAC System Time Nanoseconds Update register before writing 1 to MAC_Timestamp_Control[TSINIT] and MAC_Timestamp_Control[TSUPDT].

Diagram



Fields

Field	Function
31-0 TSS	Timestamp Seconds Indicates the timestamp seconds value. The value in this field is the seconds part of the update. • If MAC_System_Time_Nanoseconds_Update[ADDSUB] = 0, this field must be programmed with the seconds part of the update value. • If MAC_System_Time_Nanoseconds_Update[ADDSUB] = 1, this field must be programmed with the complement of the seconds part of the update value. For example, if 2.000000001 seconds need to be subtracted from the system time, MAC_System_Time_Seconds[TSS] must be FFFF_FFFEh (that is, 2^32 - 2).

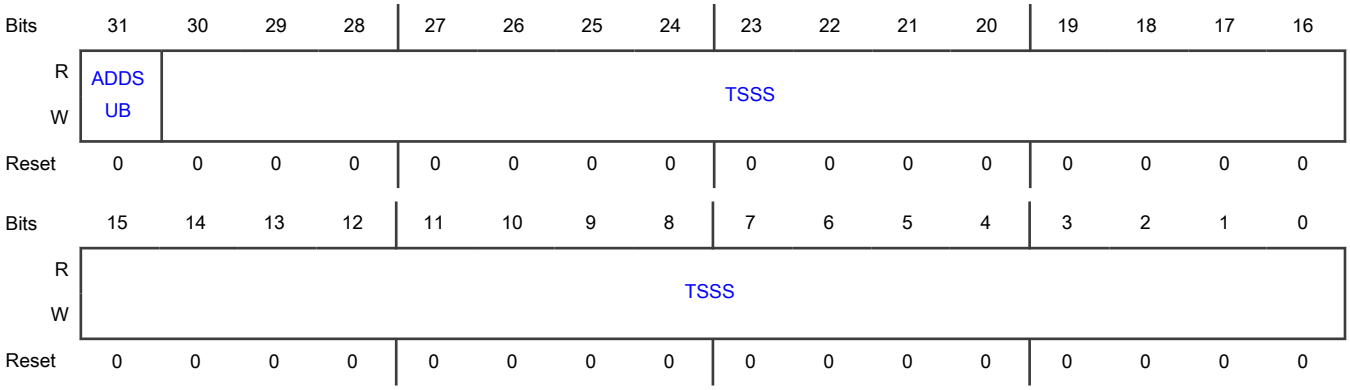
72.18.153 MAC System Time Nanoseconds Update (MAC_System_Time_Nanoseconds_Update)

Offset

Register	Offset
MAC_System_Time_Nanoseconds_Update	B14h

Function
Indicates system time nanoseconds update.

Diagram



Fields

Field	Function
31 ADDSUB	Add Or Subtract Time Indicates whether the time value is added or subtracted from the contents of the update register. • If this field is 1, the time value is subtracted from the contents of the update register. • If this field is 0, the time value is added to the contents of the update register. 0b - Add time 1b - Subtract time
30-0 TSSS	Timestamp Subseconds Indicates the sub-second part of the update. • If ADDSUB = 0, this field must be programmed with the sub-second part of the update value, with an accuracy based on MAC_Timestamp_Control[TSCTRLSSR]. • If ADDSUB = 0, this field must be programmed with the complement of the sub-second part of the update value as described below. • If MAC_Timestamp_Control[TSCTRLSSR] = 1, the programmed value must be 10^9 - <sub-second value>, and if MAC_Timestamp_Control[TSCTRLSSR] = 0, the programmed value must be 2^31 - <sub-second_value>.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If MAC_Timestamp_Control[TSCTRLSSR] = 0, each bit of it represents an accuracy of 0.46 ns, and if MAC_Timestamp_Control[TSCTRLSSR] = 1, each bit represents 1 ns and the programmed value must not exceed 3B9A_C9FFh. For example, if 2.000000001 seconds need to be subtracted from the system time, this field must be 7FFF_FFFFh (that is, 2^31 - 1) if MAC_Timestamp_Control[TSCTRLSSR] = 0. This field must be 3B9A_C9FFh (that is, 10^9 - 1) if MAC_Timestamp_Control[TSCTRLSSR] = 1.

72.18.154 MAC Timestamp Addend (MAC_Timestamp_Addend)

Offset

Register	Offset
MAC_Timestamp_Addend	B18h

Function

Is used only when the system time is configured for Fine Update mode (MAC_Timestamp_Control[TSINIT] = 1). The contents of this register are added to a 32-bit accumulator in every clock cycle (of clk_ptp_ref_i), and the system time is updated whenever the accumulator overflows.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSAR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSAR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSAR	Timestamp Addend Register Indicates the 32-bit time value to be added to the Accumulator register to achieve time synchronization.

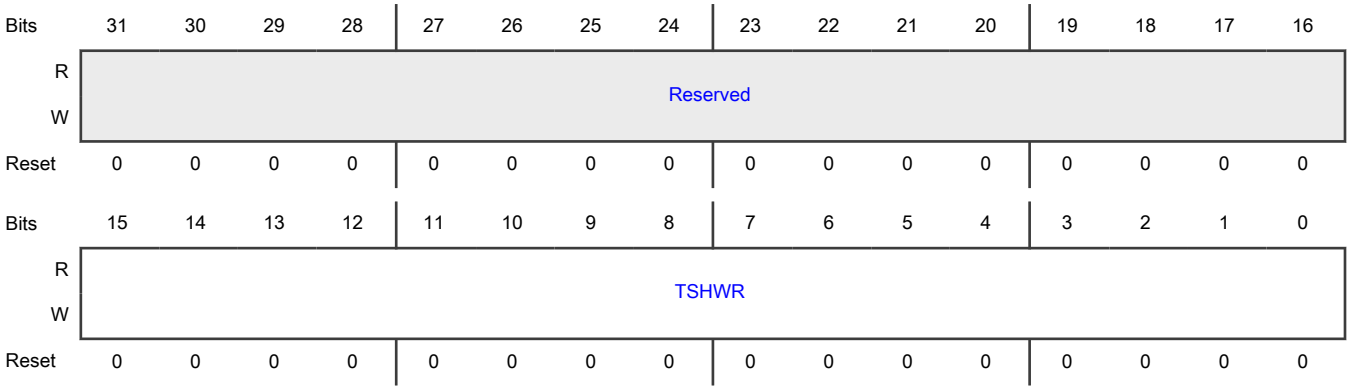
72.18.155 MAC System Time Higher Word In Seconds (MAC_System_Time_Higher_Word_Seconds)

Offset

Register	Offset
MAC_System_Time_Higher_Word_Seconds	B1Ch

Function
Indicates system time - higher word in seconds.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TSHWR	Timestamp Higher Word Register Contains the most-significant 16 bits of the timestamp seconds value. This register is optional, and you can add it if you select the IEEE 1588 higher-word register option. You write to this register directly to initialize the value, which increments when there is an overflow from the 32 bits of MAC System Time In Seconds (MAC_System_Time_Seconds). Access restrictions apply to this field, which updates based on an event occurrence.

72.18.156 MAC Timestamp Status (MAC_Timestamp_Status)

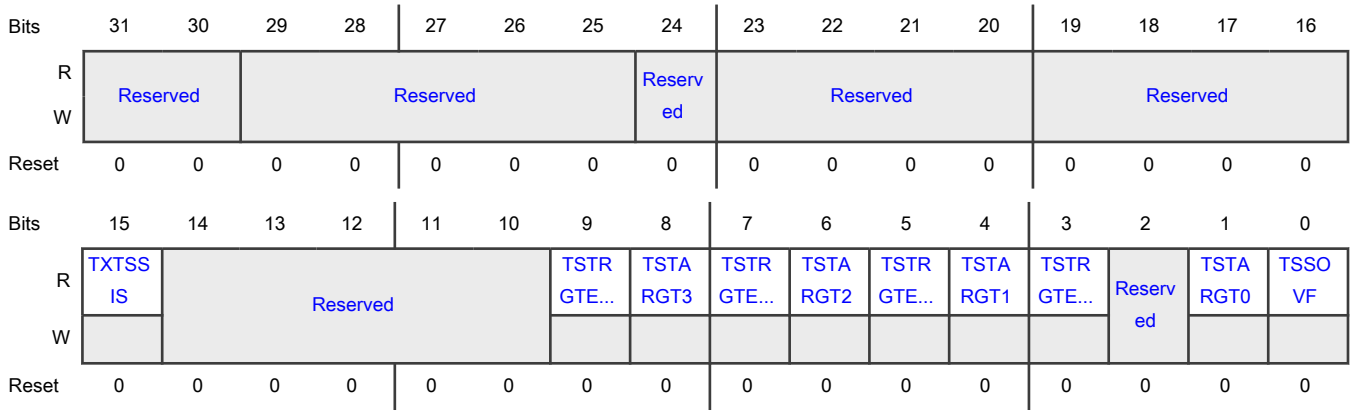
Offset

Register	Offset
MAC_Timestamp_Status	B20h

Function

Indicates timestamp status. All the bits except [27:25] are cleared when the application reads this register.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-25 —	Reserved
24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15 TXTSSIS	Transmit Timestamp Status Interrupt Status Indicates whether the transmit timestamp interrupt status is detected. In non-EQOS_CORE configurations, if the drop transmit status is enabled in MTL, this field becomes 1 when the captured transmit timestamp is updated in MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds) and MAC Transmit Timestamp Status In Nanoseconds (MAC_Tx_Timestamp_Status_Nanoseconds). If MAC_HW_Feature1[PTOEN] = 1, TXTSSIS becomes 1 when the captured transmit timestamp is updated in MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds) and MAC Transmit Timestamp Status In Nanoseconds (MAC_Tx_Timestamp_Status_Nanoseconds) for PTO-generated delay request and Pdelay request packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 0 if you read or write to MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds) when MAC_CSR_SW_Ctrl[RCWE] = 1. 0b - Not detected 1b - Detected
14-10 —	Reserved
9 TSTRGTERR3	Timestamp Target Time Error Indicates whether the timestamp target time error status is detected. This field becomes 1 if the latest target time programmed in MAC PPS3 Target Time In Seconds (MAC_PPS3_Target_Time_Seconds) and MAC PPS3 Target Time In Nanoseconds (MAC_PPS3_Target_Time_Nanoseconds) elapses. The field becomes 0 when the application reads it. Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
8 TSTARGT3	Timestamp Target Time Reached For Target Time PPS3 Indicates whether the timestamp target time reached for target time PPS3 status is detected. If this field is 1, it indicates that the value of the system time is greater than or equal to the value specified in MAC PPS3 Target Time In Seconds (MAC_PPS3_Target_Time_Seconds) and MAC PPS3 Target Time In Nanoseconds (MAC_PPS3_Target_Time_Nanoseconds). Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
7 TSTRGTERR2	Timestamp Target Time Error Indicates whether the timestamp target time error status is detected. This field becomes 1 when the latest target time programmed in MAC PPS2 Taget Time In Seconds (MAC_PPS2_Target_Time_Seconds) and MAC PPS2 Target Time In Nanoseconds (MAC_PPS2_Target_Time_Nanoseconds) elapses. The field becomes 0 when the application reads it. Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
6 TSTARGT2	Timestamp Target Time Reached For Target Time PPS2 Indicates whether the timestamp target time reached for the target time PPS2 status is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 1, it indicates that the value of system time is greater than or equal to the value specified in MAC PPS2 Taget Time In Seconds (MAC_PPS2_Target_Time_Seconds) and MAC PPS2 Target Time In Nanoseconds (MAC_PPS2_Target_Time_Nanoseconds). Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
5 TSTRGTERR1	Timestamp Target Time Error Indicates whether the timestamp target time error status is detected. This field becomes 1 when the latest target time programmed in MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds) and MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds) elapses. The field becomes 0 when the application reads it. Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
4 TSTARGT1	Timestamp Target Time Reached For Target Time PPS1 Indicates whether the timestamp target time reached for target time PPS1 status is detected. If this field is 1, it indicates that the value of system time is greater than or equal to the value specified in MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds) and MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds) Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
3 TSTRGTERR0	Timestamp Target Time Error Indicates whether the timestamp target time error status is detected. This field becomes 1 when the latest target time programmed in MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) and MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds) elapses. The field becomes 0 when the application reads it. Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 TSTARGT0	Timestamp Target Time Reached Indicates whether the timestamp target time reached status is detected. If this field is 1, it indicates that the value of the system time is greater than or equal to the value specified in MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) and MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds) Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
0 TSSOVF	Timestamp Seconds Overflow Indicates whether the timestamp seconds overflow status is detected. If this field is 1, it indicates that the seconds value of the timestamp (when supporting version 2 format) has overflowed beyond 32'hFFFF_FFFF. Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

72.18.157 MAC Transmit Timestamp Status In Nanoseconds (MAC_Tx_Timestamp_Status_Nanoseconds)

Offset

Register	Offset
MAC_Tx_Timestamp_Status_Nanoseconds	B30h

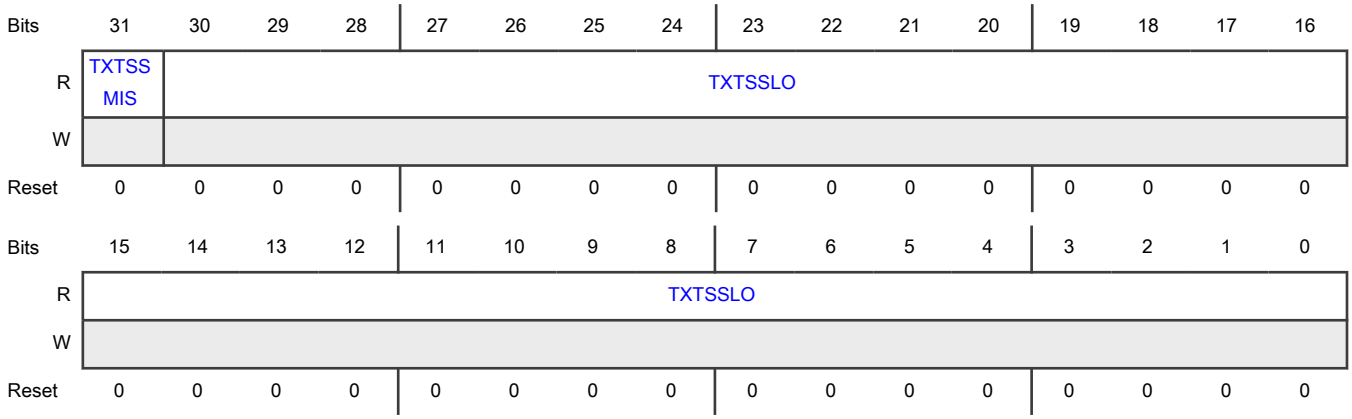
Function

Contains the nanosecond part of the timestamp captured for tansmit packets when the transmit status is disabled.

This register, along with MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds), gives the 64-bit timestamp captured for the PTP packet that MAC successfully transmits. The application considers reading this value after the last byte of this register is read. In Little-Endian mode, this means when bits [31:24] are read, and in Big-Endian mode, this means when bits [7:0] are read.

If the application does not read these registers and the timestamp of another packet is captured, then either the current timestamp is lost (overwritten) or the new timestamp is lost (dropped), depending on the settings specified in MAC_Timestamp_Control[TXTSSTSM]. The status bit TXTSC bit [15] in the MAC_Timestamp_Status register is set whenever the MAC transmitter captures the timestamp.

Diagram



Fields

Field	Function
31 TXTSSMIS	Transmit Timestamp Status Missed Indicates whether the transmit timestamp missed status is detected. If this field is 1, it indicates one of these conditions: • The timestamp of the current packet is ignored if MAC_Timestamp_Control[TXTSSTSM] = 0. • The timestamp of the previous packet is overwritten with the timestamp of the current packet if MAC_Timestamp_Control[TXTSSTSM] = 1. Access restrictions apply to this field that cleas on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
30-0 TXTSSLO	Transmit Timestamp Status Low Contains 31 bits of the Nanoseconds field of the transmit packet's captured timestamp.

72.18.158 MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds)

Offset

Register	Offset
MAC_Tx_Timestamp_Status_Seconds	B34h

Function

Contains the higher 32 bits of the timestamp (in seconds) captured when a PTP packet is transmitted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXTSSHI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXTSSHI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXTSSHI	Transmit Timestamp Status High Contains the lower 32 bits of the Seconds field of the transmitted packet's captured timestamp.

72.18.159 MAC Timestamp Ingress Asymmetry Correction (MAC_Timestamp_Ingress_Asym_Corr)

Offset

Register	Offset
MAC_Timestamp_Ingress_Asym_Corr	B50h

Function

Contains the ingress asymmetry correction value to be used when updating the correction field in the PDelay_Resp PTP messages.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OSTIAC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OSTIAC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OSTIAC	One-Step Timestamp Ingress Asymmetry Correction Contains the ingress path asymmetry value to be added to the correctionField of the Pdelay_Resp PTP packet. The programmed value must be expressed in units of nanoseconds and multiplied by 2^16. For example, 2.5 ns is represented as 00028000h. This value can also be negative, which is represented in the two's-complement form with bit 31 representing the sign bit.

72.18.160 MAC Timestamp Egress Asymmetry Correction (MAC_Timestamp_Egress_Asym_Corr)

Offset

Register	Offset
MAC_Timestamp_Egress_Asym_Corr	B54h

Function

Contains the egress asymmetry correction value to be used when updating the correction field in the PDelay_Req PTP messages.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	OSTEAC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OSTEAC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OSTEAC	One-Step Timestamp Egress Asymmetry Correction Contains the egress path asymmetry value to be subtracted from the correctionField of the Pdelay_Resp PTP packet. The programmed value must be the negated value in units of nanoseconds multiplied by 2^16. For example, if the required correction is +2.5 ns, the programmed value must be FFFD_8000h, which is the two's-complement of 0002_8000h (2.5 * 216). Similarly, if the required correction is -3.3 ns, the programmed value is 0003_4CCCh (3.3 * 216).

72.18.161 MAC Timestamp Ingress Correction In Nanoseconds (MAC_Timestamp_Ingress_Corr_Nanosecond)

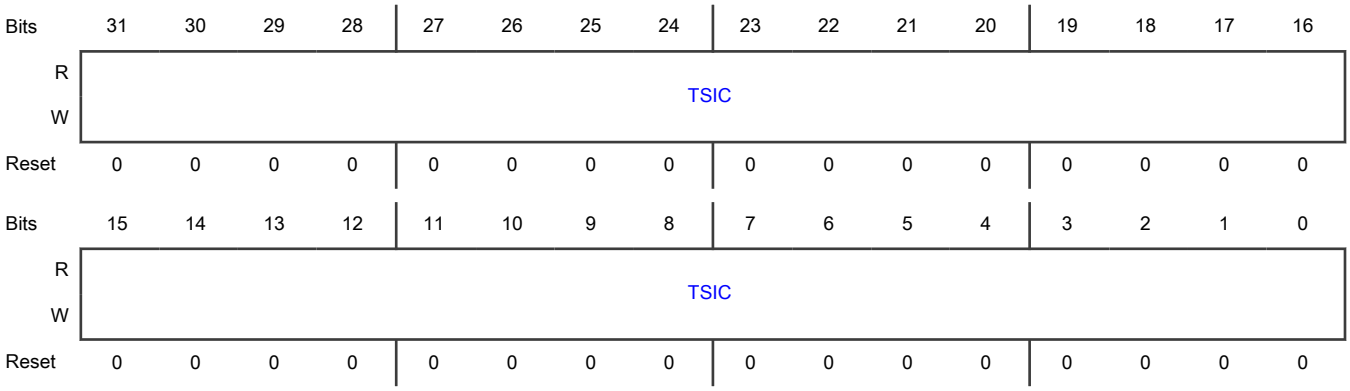
Offset

Register	Offset
MAC_Timestamp_Ingress_Corr_Nanosecond	B58h

Function

Contains the correction value, in nanoseconds, to be used with the captured timestamp value in the ingress path.

Diagram



Fields

Field	Function
31-0 TSIC	Timestamp Ingress Correction Contains the ingress path correction value as defined by the ingress correction expression.

72.18.162 MAC Timestamp Egress Correction In Nanoseconds (MAC_Timestamp_Egress_Corr_Nanosecond)

Offset

Register	Offset
MAC_Timestamp_Egress_Corr_Nanosecond	B5Ch

Function

Contains the correction value, in nanoseconds, to be used with the captured timestamp value in the egress path.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSEC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSEC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSEC	Timestamp Egress Correction Contains the nanoseconds part of the egress path correction value as defined by the egress correction expression.

72.18.163 MAC Timestamp Ingress Correction In Subnanoseconds (MAC_Timestamp_Ingress_Corr_Subnanosec)

Offset

Register	Offset
MAC_Timestamp_Ingress_Corr_Subnanosec	B60h

Function

Contains the sub-nanosecond part of the correction value to be used with the captured timestamp value for ingress direction.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSICSNS								Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 TSICSNS	Timestamp Ingress Correction In Sub-Nanoseconds Contains the sub-nanoseconds part of the ingress path correction value as defined by the Ingress Correction expression.
7-0 —	Reserved

72.18.164 MAC Timestamp Engress Correction In Subnanoseconds (MAC_Timestamp_Egress_Corr_Subnanosec)

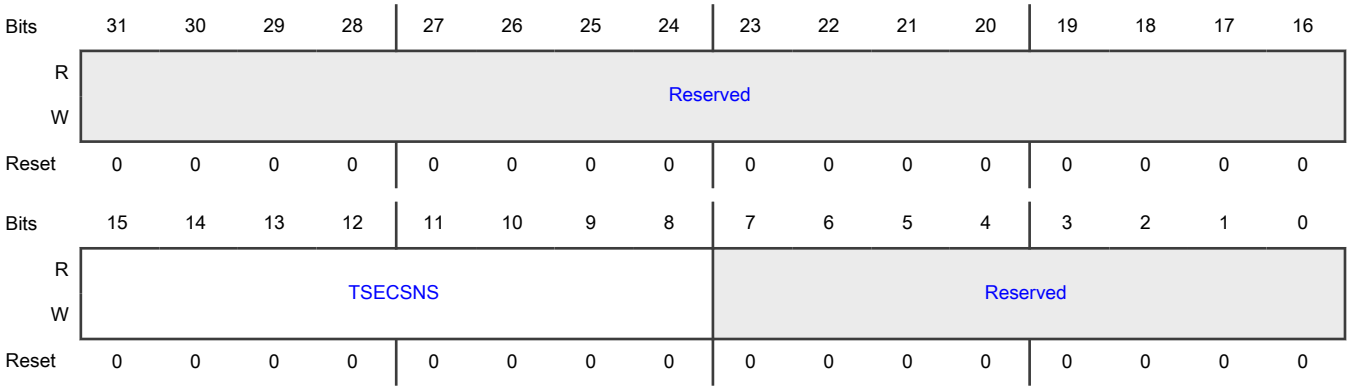
Offset

Register	Offset
MAC_Timestamp_Egress_Corr_Subnanosec	B64h

Function

Contains the sub-nanosecond part of the correction value to be used with the captured timestamp value for the egress direction.

Diagram



Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-8 TSECSNS	Timestamp Egress Correction In Sub-Nanoseconds Contains the sub-nanoseconds part of the egress path correction value, as defined by the Egress Correction expression.
7-0 —	Reserved

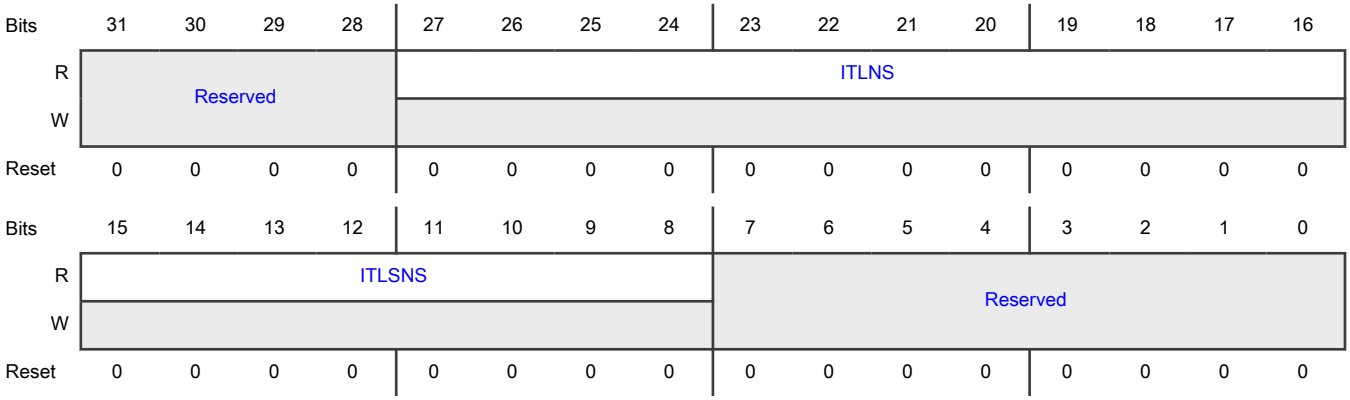
72.18.165 MAC Timestamp Ingress Latency (MAC_Timestamp_Ingress_Latency)

Offset

Register	Offset
MAC_Timestamp_Ingress_Latency	B68h

Function
Holds the ingress MAC latency.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16	Ingress Timestamp Latency In Sub-Nanoseconds

Table continues on the next page...

Table continued from the previous page...

Field	Function
ITLNS	Holds the average latency, in sub-nanoseconds, between the MAC input ports (phy_rxd_i) and the actual point (GMII/MII), where the ingress timestamp is taken. The ingress correction value is computed as described in section 7.1.2.4.1 of QoS Databook.
15-8 ITLSNS	Ingress Timestamp Latency In Nanoseconds Holds the average latency, in nanoseconds, between the MAC input ports (phy_rxd_i) and the actual point (GMII/MII), where the ingress timestamp is taken. The ingress correction value is computed as described in section 7.1.2.4.1 of QoS Databook.
7-0 —	Reserved

72.18.166 MAC Timestamp Egress Latecy (MAC_Timestamp_Egress_Latency)

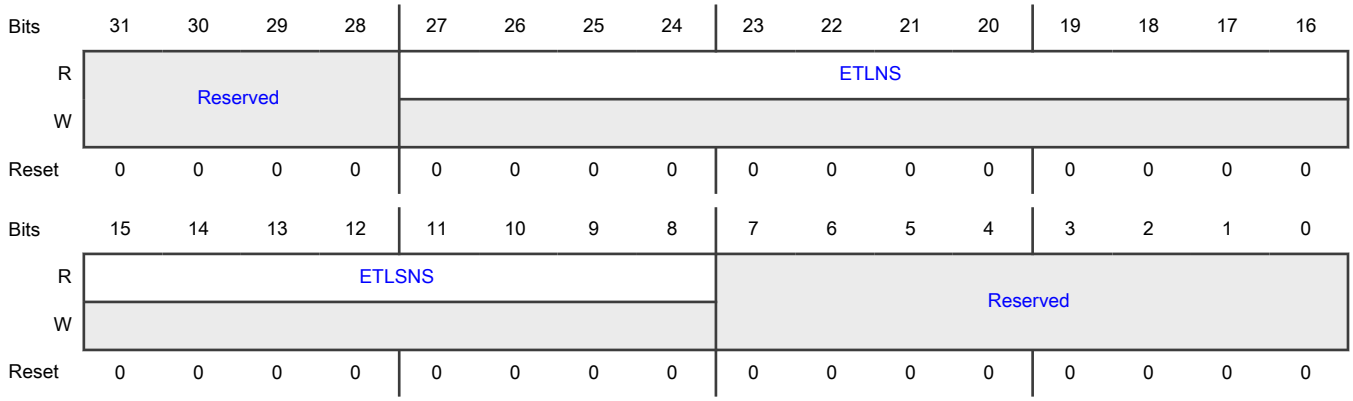
Offset

Register	Offset
MAC_Timestamp_Egress_Latency	B6Ch

Function

Holds the egress MAC latency.

Diagram



Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-16 ETLNS	Egress Timestamp Latency In Nanoseconds Holds the average latency, in nanoseconds, between the actual point (GMII/MII) where the egress timestamp is taken and the MAC output ports (phy_txd_o). The ingress correction value is computed as described in section 7.1.2.4.2 of QoS Databook.
15-8 ETLSNS	Egress Timestamp Latency In Sub-Nanoseconds Holds the average latency, in sub-nanoseconds, between the actual point (GMII/MII) where the egress timestamp is taken and the MAC output ports (phy_txd_o). The ingress correction value is computed as described in section 7.1.2.4.2 of QoS Databook.
7-0 —	Reserved

72.18.167 MAC PPS Control (MAC_PPS_Control)

Offset

Register	Offset
MAC_PPS_Control	B70h

Function

Indicates PPS control.

In this register:

- Bits [30:24] are valid only when four flexible PPS outputs are selected.
- Bits [22:16] are valid only when three or more flexible PPS outputs are selected.
- Bits [14:8] are valid only when two or more flexible PPS outputs are selected.
- Bits [6:4] are valid only when the flexible PPS feature is selected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MCGREN3	TRGTMODSEL3		Reserved	PPSCMD3				MCGREN2	TRGTMODSEL2		Reserved	PPSCMD2			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MCGREN1	TRGTMODSEL1		Reserved	PPSCMD1				MCGREN0	TRGTMODSEL0		PPSEN0	PPSCTRL_PPSCMD			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MCGREN3	MCGR Mode Enable For PPS3 Output Enables the third PPS instance to operate in PPS or MCGR mode. If this field is 1, it operates in MCGR mode, and if the value is 0, it operates in PPS mode.
30-29 TRGTMODSEL3	Target Time Register Mode For PPS3 Output Indicates MAC PPS3 Target Time In Seconds (MAC_PPS3_Target_Time_Seconds) and MAC PPS3 Target Time In Nanoseconds (MAC_PPS3_Target_Time_Nanoseconds) mode for the PPS3 output signal. 00b - Target Time registers are programmed only for generating the interrupt event. The flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding ptp_pps_o output port. 01b - Reserved 10b - Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation. 11b - Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted.
28 —	Reserved
27-24 PPSCMD3	Flexible PPS3 Output Control Controls the flexible PPS3 output (ptp_pps_o[3]) signal. The functioning of this field is similar to that of PPSCMD0[2:0] (presentation time control bits). If MCGREN3 = 1, PPSCMD3 that includes bits [27:24] are considered as presentation time control bits for media clock generation and recovery for comparator instance 3. If MCGREN3 is not 1, only three bits [26:24] are considered as PPSCMD3 and the fourth bit becomes 0. Access restrictions apply to this field that clears automatically. Writing 0 to it has no effect.
23 MCGREN2	MCGR Mode Enable For PPS2 Output Enables or disables the second PPS instance to operate in PPS or MCGR mode. If this field is 1, it operates in MCGR mode, and if the value is 0, it operates in PPS mode. 0b - Disabled 1b - Enabled
22-21 TRGTMODSEL2	Target Time Register Mode For PPS2 Output Indicates MAC PPS2 Taget Time In Seconds (MAC_PPS2_Target_Time_Seconds) and MAC PPS2 Target Time In Nanoseconds (MAC_PPS2_Target_Time_Nanoseconds) mode for the PPS2 output signal. 00b - Target Time registers are programmed only for generating the interrupt event. The flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding ptp_pps_o output port. 01b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation. 11b - Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted.
20 —	Reserved
19-16 PPSCMD2	Flexible PPS2 Output Control Controls the flexible PPS2 output (ptp_pps_o[2]) signal. The functioning of this field is similar to that of the PPSCMD0 (presentation time control bits). If MCGREN2 = 1, the bits included in PPSCMD2 [19:16] are considered as presentation time control bits for media clock generation and recovery for comparator instance 2. If MCGREN2 is not 1, only three bits [18:16] are used as PPSCMD2 and the fourth bit becomes 0. Access restrictions apply to this field that clears automatically. Writing 0 to it has no effect.
15 MCGREN1	MCGR Mode Enable For PPS1 Output Enables or disables the first PPS instance to operate in PPS or MCGR mode. If this field is 1, it operates in MCGR mode, and if the value is 0, it operates in PPS mode. 0b - Disabled 1b - Enabled
14-13 TRGTMODSEL1	Target Time Register Mode For PPS1 Output Indicates MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds) and MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds) mode for the PPS1 output signal. 00b - Target Time registers are programmed only for generating the interrupt event. The flexible PPS function must not be 1 in this mode, otherwise spurious transitions may be observed on the corresponding ptp_pps_o output port. 01b - Reserved 10b - Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation. 11b - Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted.
12 —	Reserved
11-8 PPSCMD1	Flexible PPS1 Output Control Controls the flexible PPS1 output (ptp_pps_o[1]) signal. The functioning of this field is similar to that of the PPSCMD0 field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If MCGREN1 = 1, the bits included in PPSCMD1 [11:8] are considered as presentation time control bits for media clock generation and recovery for comparator instance 1. The functioning of PPSCMD1 is similar to that of the PPSCMD0 presentation time control bits. If MCGREN1 is not 1, only three bits [10:8] are used as PPSCMD1 and the fourth bit becomes 0. Access restrictions apply to this field that clears automatically. Writing 0 to it has no effect.
7 MCGREN0	MCGR Mode Enable For PPS0 Output Indictaes whether the 0th PPS instance is enabled to operate in PPS or MCGR mode. If this field is 1, it operates in MCGR mode, and if the value is 0, it operates in PPS mode. 0b - PPS mode 1b - MCGR mode
6-5 TRGTMODSEL0	Target Time Register Mode For PPS0 Output Indicates MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) and MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds) mode for the PPS0 output signal. 00b - Target Time registers are programmed only for generating the interrupt event. The flexible PPS function must not be 1 in this mode, otherwise spurious transitions may be observed on the corresponding ptp_pps_o output port. 01b - Reserved 10b - Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation. 11b - Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted.
4 PPSEN0	Flexible PPS Output Mode Enable 0 Indicates the status of Flexible PPS Output mode. • If this field is 1, the PPSCTRL_PPSCMD field functions as PPSCMD. • If this field is 0, the PPSCTRL_PPSCMD field functions as PPSCTRL (fixed PPS mode). 0b - Disabled 1b - Enabled
3-0 PPSCTRL_PPSCMD	PPS Output Frequency Control Controls the frequency of the PPS0 output (ptp_pps_o[0] signal). The default value of this field is 0, and the PPS output is 1 pulse (of width clk_ptp_i) every second. For the other values of PPSCTRL, the PPS output becomes a generated clock having these frequencies: • 0001: The binary rollover is 2 Hz, and the digital rollover is 1 Hz. • 0010: The binary rollover is 4 Hz, and the digital rollover is 2 Hz. • 0011: The binary rollover is 8 Hz, and the digital rollover is 4 Hz. • 0100: The binary rollover is 16 Hz, and the digital rollover is 8 Hz.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• .. • 1111: The binary rollover is 32.768 kHz and the digital rollover is 16.384 kHz. **NOTE** In Binary Rollover mode, the PPS output (ptp_pps_o) has a duty cycle of 50 percent with these frequencies. In Digital Rollover mode, the PPS output frequency is an average number. The actual clock has a different frequency that is synchronized every second. For example, • If PPSCTRL = 0001, PPS (1 Hz) has a low period of 537 ms and a high period of 463 ms. • If PPSCTRL = 0010, PPS (2 Hz) is a sequence of one clock of 50 percent duty cycle and a 537 ms period. The second clock has a period of 463 ms (268 ms low and 195 ms high). • If PPSCTRL = 0011, PPS (4 Hz) is a sequence of three clocks of 50 percent duty cycle and a 268 ms period, and the fourth clock of 195 ms period (134 ms low and 61 ms high). This behavior is a result of the non-linear toggling of bits in Digital Rollover mode in MAC System Time In Nanoseconds (MAC_System_Time_Nanoseconds) or flexible PPS output (ptp_pps_o[0]) control. Programming these bits with a non-zero value instructs MAC to initiate an event. When the command is transferred or synchronized with the PTP clock domain, these bits automatically become 0. You must ensure that these bits are programmed only when they are "all-zero." The following list describes the values of PPSCMD0. • 0000: No command • 0001: Start a single pulse. This command generates a single pulse rising at the start point defined in MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) and MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds), for a duration defined in MAC PPS0 Width (MAC_PPS0_Width). • 0010: Start pulse train. This command generates the train of pulses rising at the start point defined in the Target Time registers and for a duration defined in MAC PPS0 Width (MAC_PPS0_Width), repeated at an interval defined in the PPS Interval registers. By default, the PPS pulse train runs freely unless the "stop pulse train at time" or "stop pulse train immediately" command stops it. • 0011: Cancel start. This command cancels the start single pulse and start pulse train commands if the system time has not crossed the programmed start time. • 0100: Stop pulse train at time. This command stops the train of pulses that the start pulse train command (PPSCMD = 0010) initiates after the time programmed in the Target Time registers elapses. • 0101: Stop pulse train immediately. This command immediately stops the train of pulses that the start pulse train command (PPSCMD = 0010) initiates. • 110: Cancel stop pulse train. This command cancels the stop pulse train at the time command if the programmed stop time has not elapsed. The PPS pulse train runs freely if the command executes successfully. • 0111-1111: Reserved or presentation time control. If MAC_PPS_Control[MCGREN0] = 0, then these are treated as presentation time control bits. This list describes the values of PPSCMD0:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0000: The MCGR operation is not carried out. If set to this value in the mid of clock recovery or generation, all the processing inputs are flushed. • 0001: Captures the presentation time at the rising edge of mcg_pst_trig_i[0] in MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) • 0010: Captures the presentation time at the falling edge of mcg_pst_trig_i[0] in MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) • 0011: Captures the presentation time at both the edges of mcg_pst_trig_i[0] in MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) • 0100-1000: Reserved • 1001: Toggle output on compare • 1010: Pulse output low on compare for one PTP-clock cycle • 1011: Pulse output high on compare for one PTP-clock cycle • 1100-1111: Reserved

72.18.168 MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds)

Offset

Register	Offset
MAC_PPS0_Target_Time_Seconds	B80h

Function

Is used, along with the PPS Target Time Nanoseconds register, to schedule an interrupt event (MAC_Timestamp_Status[TSTARGT0]) when the system time exceeds the value programmed in these registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSTRH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSTRH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSTRH0	PPS Target Time In Seconds Register Stores the target time in seconds. When the timestamp value matches or exceeds both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt (if enabled), based on Target Time mode selected for the corresponding PPS output in MAC PPS Control (MAC_PPS_Control). If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and MAC_Timestamp_Control[PTGE] = 1, with the presentation time control set in Recovery mode, these fields indicate that the application programs the TPT. In Generation mode, it indicates that CPT is generated at the sampled trigger.

72.18.169 MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds)

Offset

Register	Offset
MAC_PPS0_Target_Time_Nanoseconds	B84h

Function
Indicates the target time in nanoseconds for PPS0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRGTBUS...				TTSL0											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TTSL0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TRGTBUSY0	PPS Target Time Busy Status 0 Indicates whether the PPS target time busy status 0 is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• MAC writes 1 to this field when MAC_PPS_Control[PPSCTRL_PPSCMD] is programmed to 010 or 011. This instructs MAC to synchronize the Target Time registers with the PTP clock domain. • MAC writes 0 to this field after synchronizing the Target Time registers with the PTP clock domain. The application must not update the Target Time registers when this field is read as 1. Otherwise, the synchronization of the previously programmed time is corrupted. 0b - Not detected 1b - Detected
30-0 TTSL0	Target Time Low For PPS0 Stores the time in (signed) nanoseconds. If the value of the timestamp matches the value in both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on MAC_PPS_Control[TRGTMODSEL0]. • If MAC_Timestamp_Control[TSCTRLSSR] = 0, this value must be calculated as (time in ns / 0.465). The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. • If MAC_Timestamp_Control[TSCTRLSSR] = 1, this value must not exceed 3B9A_C9FFh. The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. Access restriction apply to this field that clears automatically. Writing 0 to it has no effect.

72.18.170 MAC PPS0 Interval (MAC_PPS0_Interval)

Offset

Register	Offset
MAC_PPS0_Interval	B88h

Function

Contains the number of units of the sub-second increment value between the rising edges of the PPS0 signal output (ptp_pps_o[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSINT0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSINT0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSINT0	PPS Output Signal Interval 0 Stores the interval between the rising edges of the PPS0 signal output. The interval is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and the desired interval between the rising edges of the PPS0 signal output is 100 ns (that is, five units of the sub-second increment value), you must program value 4 (5-1) in this register.

72.18.171 MAC PPS0 Width (MAC_PPS0_Width)

Offset

Register	Offset
MAC_PPS0_Width	B8Ch

Function

Contains the number of units of the sub-second increment value between the rising and corresponding falling edges of the PPS0 signal output (ptp_pps_o[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSWIDTH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSWIDTH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSWIDTH0	PPS Output Signal Width 0 Stores the width between the rising and corresponding falling edges of the PPS0 signal output. The width is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock has a frequency of 50 MHz (period of 20 ns), and a width between the rising and corresponding falling edges of the PPS0 signal output is 80 ns (that is, four units of the sub-second increment value), you should program value 3 (4-1) in this register. **NOTE** The value programmed in this register must be lesser than the value programmed in MAC PPS0 Interval (MAC_PPS0_Interval).

72.18.172 MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds)

Offset

Register	Offset
MAC_PPS1_Target_Time_Seconds	B90h

Function
Is used to schedule an interrupt event (MAC_Timestamp_Status[TSTARGT0]), along with the PPS Target Time Nanoseconds register, when the system time exceeds the value programmed in these registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSTRH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSTRH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0	PPS Target Time In Seconds 1

Table continues on the next page...

Field	Function
TSTRH1	Stores the target time in seconds. When the timestamp value matches or exceeds both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on Target Time mode selected for the corresponding PPS output in MAC PPS Control (MAC_PPS_Control). If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and MAC_Timestamp_Control[PTGE] = 1, with the presentation time control set in Recovery mode, these fields indicate that the application programs the TPT. In Generation mode, it indicates that CPT is generated at the sampled trigger.

72.18.173 MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds)

Offset

Register	Offset
MAC_PPS1_Target_Time_Nanoseconds	B94h

Function
Indicates the target time in nanoseconds for PPS1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRGTBUS...				TTSL1											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TTSL1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TRGTBUSY1	PPS Target Time Busy Status 1 Indicates whether the PPS target time busy status 1 is detected. • MAC writes 1 to this field when MAC_PPS_Control[PPSCTRL_PPSCMD] is programmed to 010 or 011. Programming the PPSCMD0 field as 010 or 011 instructs MAC to synchronize the Target Time registers with the PTP clock domain.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• MAC writes 0 to this field after synchronizing the Target Time registers with the PTP clock domain. The application must not update the Target Time registers when this field is read as 1. Otherwise, the synchronization of the previously programmed time is corrupted. 0b - Not detected 1b - Detected
30-0 TTSL1	Target Time Low For PPS1 Stores the time in (signed) nanoseconds. If the value of the timestamp matches the value in both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on MAC_PPS_Control[TRGTMODSEL0]. • If MAC_Timestamp_Control[TSCTRLSSR] = 0, this value must be defined as (time in ns / 0.465). The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. • If MAC_Timestamp_Control[TSCTRLSSR] = 1, this value must not exceed 3B9A_C9FFh. The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. Access restriction apply to this field that clears automatically. Writing 0 to it has no effect.

72.18.174 MAC PPS1 Interval (MAC_PPS1_Interval)

Offset

Register	Offset
MAC_PPS1_Interval	B98h

Function

Contains the number of units of the sub-second increment value between the rising edges of the PPS0 signal output (ptp_pps_o[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSINT1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSINT1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSINT1	PPS Output Signal Interval 1 Stores the interval between the rising edges of the PPS0 signal output. The interval is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and the desired interval between the rising edges of the PPS0 signal output is 100 ns (that is, five units of the sub-second increment value), you must program value 4 (5-1) in this register.

72.18.175 MAC PPS1 Width (MAC_PPS1_Width)

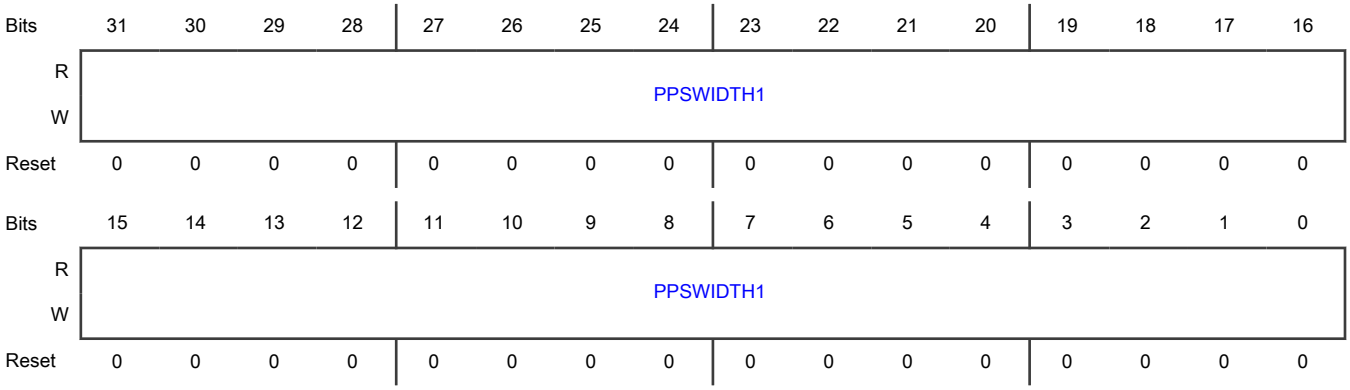
Offset

Register	Offset
MAC_PPS1_Width	B9Ch

Function

Contains the number of units of the sub-second increment value between the rising and corresponding falling edges of the PPS0 signal output (ptp_pps_o[0]).

Diagram



Fields

Field	Function
31-0 PPSWIDTH1	PPS Output Signal Width 1 Stores the width between the rising and corresponding falling edges of the PPS0 signal output. The width is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock has a frequency of 50 MHz (period of 20 ns), and a width between the rising and corresponding falling edges of the PPS0 signal output is 80 ns (that is, four units of the sub-second increment value), you should program value 3 (4-1) in this register.

Table continues on the next page...

Field	Function
	NOTE The value programmed in this register must be lesser than the value programmed in MAC PPS0 Interval (MAC_PPS0_Interval).

72.18.176 MAC PPS2 Taget Time In Seconds (MAC_PPS2_Target_Time_Seconds)

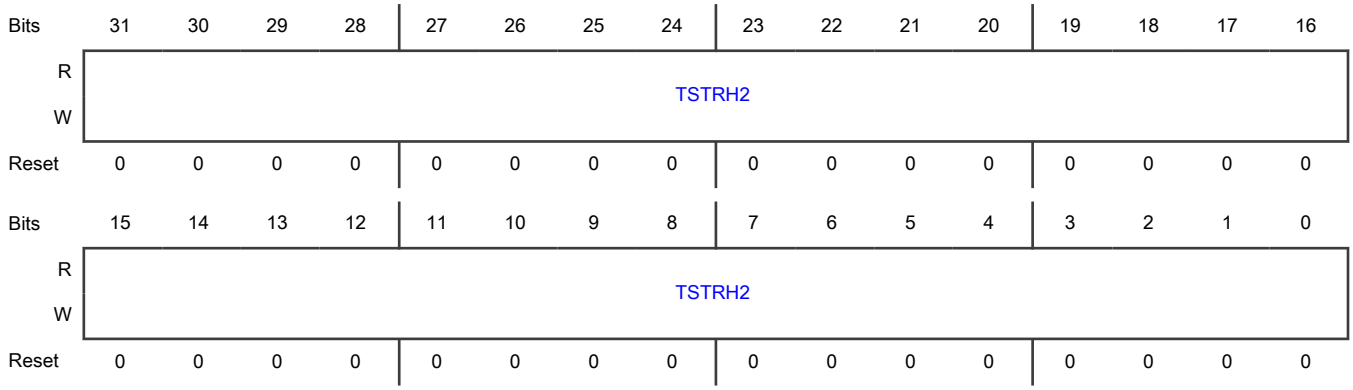
Offset

Register	Offset
MAC_PPS2_Target_Time_Seconds	BA0h

Function

Is used to schedule an interrupt event (MAC_Timestamp_Status[TSTARGT0]), along with the PPS Target Time Nanoseconds register, when the system time exceeds the value programmed in these registers.

Diagram



Fields

Field	Function
31-0 TSTRH2	PPS Target Time In Seconds 2 Stores the target time in seconds. When the timestamp value matches or exceeds both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on Target Time mode selected for the corresponding PPS output in MAC PPS Control (MAC_PPS_Control). If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and MAC_Timestamp_Control[PTGE] = 1, with the presentation time control set in Recovery mode, these fields indicate that the application programs the TPT. In Generation mode, it indicates that CPT is generated at the sampled trigger.

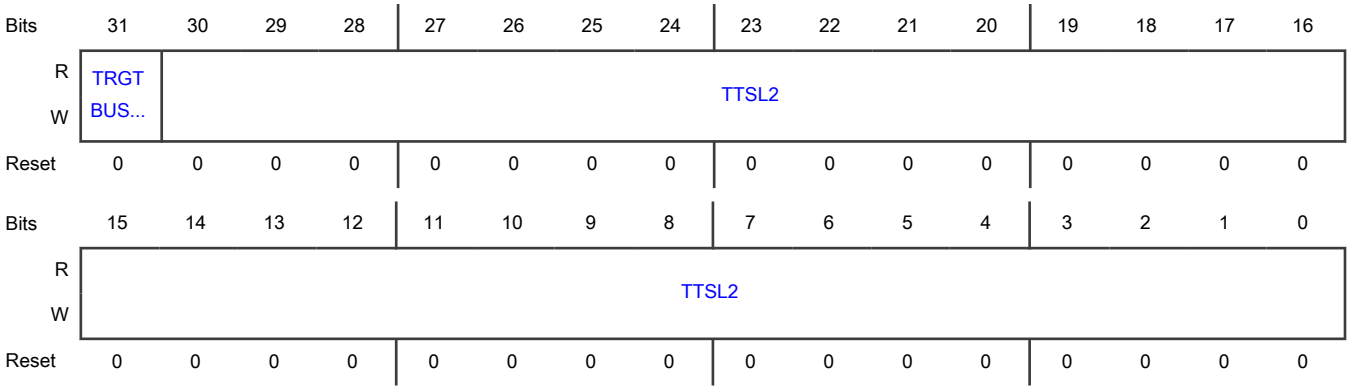
72.18.177 MAC PPS2 Target Time In Nanoseconds (MAC_PPS2_Target_Time_Nanoseconds)

Offset

Register	Offset
MAC_PPS2_Target_Time_Nanoseconds	BA4h

Function
Indicates the target time in nanoseconds for PPS2.

Diagram



Fields

Field	Function
31 TRGTBUSY2	PPS Target Time Busy Status 2 Indicates whether the PPS target time busy status 2 is detected. • MAC writes 1 to this field when MAC_PPS_Control[PPSCTRL_PPSCMD] is programmed to 010 or 011. This instructs MAC to synchronize the Target Time registers with the PTP clock domain. • MAC writes 0 to this field after synchronizing the Target Time registers with the PTP clock domain. The application must not update the Target Time registers when this field is read as 1. Otherwise, the synchronization of the previously programmed time is corrupted. 0b - Not detected 1b - Detected
30-0 TTSL2	Target Time Low For PPS2 Stores the time in (signed) nanoseconds. If the value of the timestamp matches the value in both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on MAC_PPS_Control[TRGTMODSEL0].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If MAC_Timestamp_Control[TSCTRLSSR] = 0, this value must be defined as (time in ns / 0.465). The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. • If MAC_Timestamp_Control[TSCTRLSSR] = 1, this value must not exceed 3B9A_C9FFh. The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. Access restriction apply to this field that clears automatically. Writing 0 to it has no effect.

72.18.178 MAC PPS2 Interval (MAC_PPS2_Interval)

Offset

Register	Offset
MAC_PPS2_Interval	BA8h

Function

Contains the number of units of the sub-second increment value between the rising edges of the PPS0 signal output (ptp_pps_o[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSINT2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSINT2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSINT2	PPS Output Signal Interval 2 Stores the interval between the rising edges of the PPS0 signal output. The interval is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and the desired interval between the rising edges of the PPS0 signal output is 100 ns (that is, five units of the sub-second increment value), you must program value 4 (5-1) in this register.

72.18.179 MAC PPS2 Width (MAC_PPS2_Width)

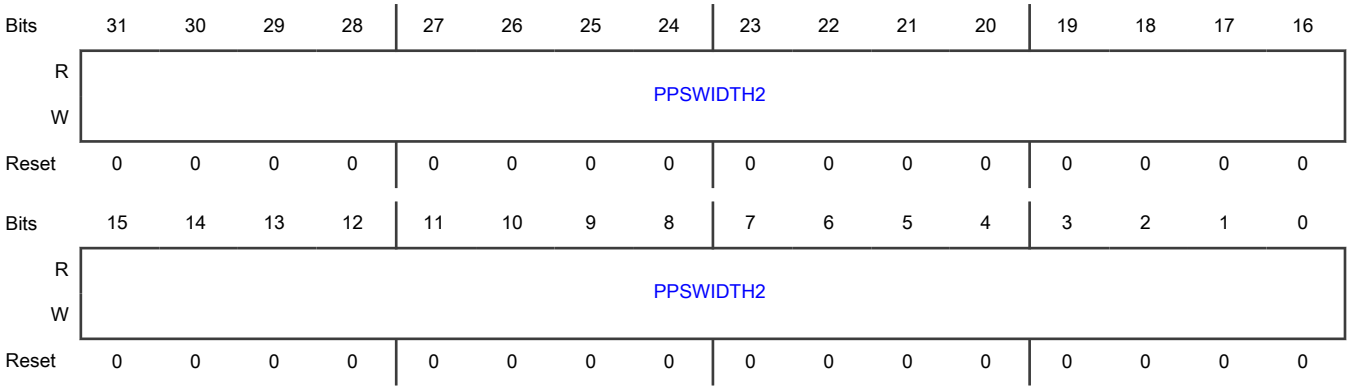
Offset

Register	Offset
MAC_PPS2_Width	BACh

Function

Contains the number of units of the sub-second increment value between the rising and corresponding falling edges of the PPS0 signal output (ptp_pps_o[0]).

Diagram



Fields

Field	Function
31-0 PPSWIDTH2	PPS Output Signal Width 2 Stores the width between the rising and corresponding falling edges of the PPS0 signal output. The width is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock has a frequency of 50 MHz (period of 20 ns), and a width between the rising and corresponding falling edges of the PPS0 signal output is 80 ns (that is, four units of the sub-second increment value), you should program value 3 (4-1) in this register. **NOTE** The value programmed in this register must be lesser than the value programmed in MAC PPS0 Interval (MAC_PPS0_Interval).

72.18.180 MAC PPS3 Target Time In Seconds (MAC_PPS3_Target_Time_Seconds)

Offset

Register	Offset
MAC_PPS3_Target_Time_Seconds	BB0h

Function

Is used to schedule an interrupt event (MAC_Timestamp_Status[TSTARGT0]), along with the PPS Target Time Nanoseconds register, when the system time exceeds the value programmed in these registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSTRH3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSTRH3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSTRH3	PPS Target Time In Seconds 3 Stores the target time in seconds. When the timestamp value matches or exceeds both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on Target Time mode selected for the corresponding PPS output in MAC PPS Control (MAC_PPS_Control). If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and MAC_Timestamp_Control[PTGE] = 1, with the presentation time control set in Recovery mode, these fields indicate that the application programs the TPT. In Generation mode, it indicates that CPT is generated at the sampled trigger.

72.18.181 MAC PPS3 Target Time In Nanoseconds (MAC_PPS3_Target_Time_Nanoseconds)

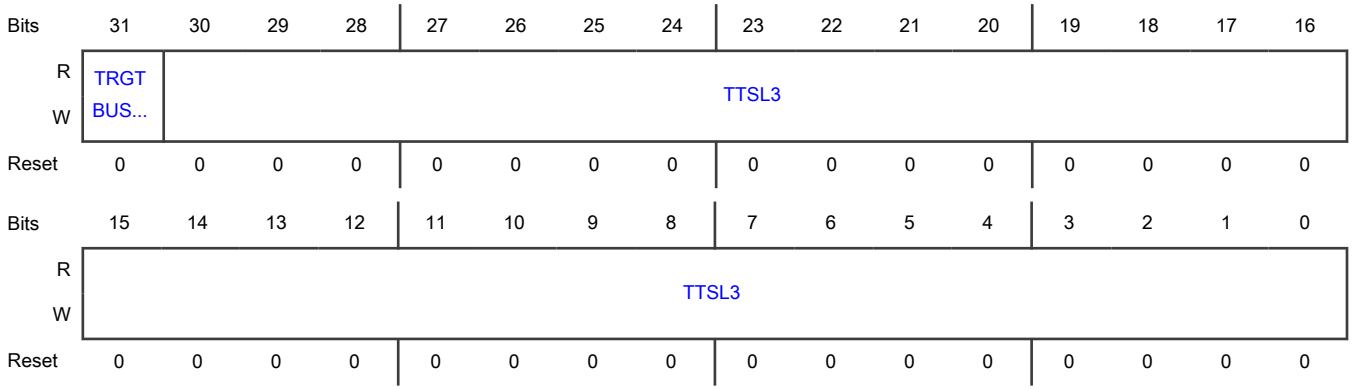
Offset

Register	Offset
MAC_PPS3_Target_Time_Nanoseconds	BB4h

Function

Indicates the target time in nanoseconds for PPS3.

Diagram



Fields

Field	Function
31 TRGTBUSY3	PPS Target Time Register Busy 3 Indicates whether the PPS target time register busy status is detected. MAC writes 1 to this field when MAC_PPS_Control[PPSCTRL_PPSCMD] is programmed to 010 or 011. This instructs MAC to synchronize the Target Time registers with the PTP clock domain. MAC writes 0 to this field after synchronizing the Target Time registers with the PTP clock domain. The application must not update the Target Time registers when this field is read as 1. Otherwise, the synchronization of the previously programmed time is corrupted. 0b - Not detected 1b - Detected
30-0 TTSL3	Target Time Low For PPS3 Stores the time in (signed) nanoseconds. If the value of the timestamp matches the value in both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on MAC_PPS_Control[TRGTMODSEL0]. • If MAC_Timestamp_Control[TSCTRLSSR] = 0, this value must be defined as (time in ns / 0.465). The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. • If MAC_Timestamp_Control[TSCTRLSSR] = 1, this value must not exceed 3B9A_C9FFh. The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. Access restriction apply to this field that clears automatically. Writing 0 to it has no effect.

72.18.182 MAC PPS3 Interval (MAC_PPS3_Interval)

Offset

Register	Offset
MAC_PPS3_Interval	BB8h

Function

Contains the number of units of the sub-second increment value between the rising edges of the PPS0 signal output (ptp_pps_o[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSINT3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSINT3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSINT3	PPS Output Signal Interval Stores the interval between the rising edges of the PPS0 signal output. The interval is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and the desired interval between the rising edges of the PPS0 signal output is 100 ns (that is, five units of the sub-second increment value), you must program value 4 (5-1) in this register.

72.18.183 MAC PPS3 Width (MAC_PPS3_Width)

Offset

Register	Offset
MAC_PPS3_Width	BBCh

Function

Contains the number of units of the sub-second increment value between the rising and corresponding falling edges of the PPS0 signal output (ptp_pps_o[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSWIDTH3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSWIDTH3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSWIDTH3	PPS Output Signal Width 3 Stores the width between the rising and corresponding falling edges of the PPS0 signal output. The width is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock has a frequency of 50 MHz (period of 20 ns), and a width between the rising and corresponding falling edges of the PPS0 signal output is 80 ns (that is, four units of the sub-second increment value), you should program value 3 (4-1) in this register. **NOTE** The value programmed in this register must be lesser than the value programmed in MAC PPS0 Interval (MAC_PPS0_Interval).

72.18.184 MTL Operation Mode (MTL_Operation_Mode)

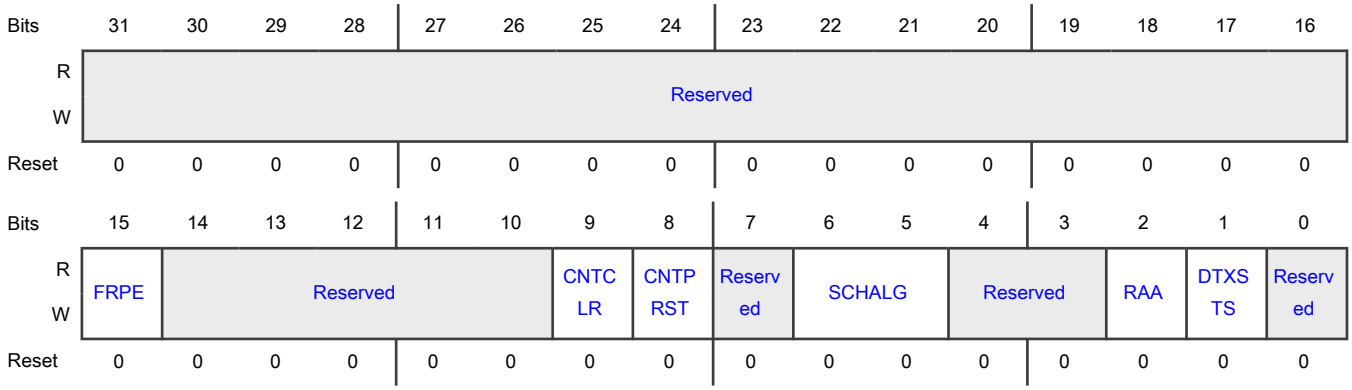
Offset

Register	Offset
MTL_Operation_Mode	C00h

Function

Establishes the transmit and receive operating modes and commands.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 FRPE	Flexible Receive Parser Enable Indicates the status of the flexible receive parser. • If this field is 1, the programmable receive parser functionality is enabled. • If the receive parser is disabled and is in the middle of parsing, the receive parser functionality is disabled only after completing the current packet parsing. When the receive parser is enabled from the disabled state, the receive parser is activated for the next immediate packet. 0b - Disabled 1b - Enabled
14-10 —	Reserved
9 CNTCLR	Counters Reset Indicates whether the counters are reset. • If this field is 1, all the counters are reset. The field cleares automatically after one clock cycle. • If you write 1 to this field along with the CNTPRST field, the CNT_PRESET field takes precedence. Access restrictions apply to this field that clears automatically. Writing 0 to it has no effect. 0b - Not reset 1b - Reset
8 CNTPRST	Counters Preset Indicates the status of preset counters. If this field is 1,

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• MTL Tx Queue 0 Underflow (MTL_TxQ0_Underflow)[bits 0-7] is initialized or preset to 12'h7F0. • Missed packet and overflow packet counters in MTL Rx Queue Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt)[bits 0-7] are initialized or preset to 12'h7F0. Access restrictions apply to this field that clears automatically. Writing 0 to it has no effect. 0b - Disabled 1b - Enabled
7 —	Reserved
6-5 SCHALG	Transmit Scheduling Algorithm Indicates the algorithm for transmit scheduling. 00b - WRR algorithm 01b - WFQ algorithm when DCB feature selected; otherwise, reserved 10b - DWRR algorithm when DCB feature selected; otherwise, reserved 11b - Strict priority algorithm
4-3 —	Reserved
2 RAA	Receive Arbitration Algorithm Is used to select the arbitration algorithm for the receive side. Queue 0 has the lowest priority and the last queue has the highest priority. 0b - Strict priority (SP) 1b - Weighted strict priority (WSP)
1 DTXSTS	Drop Transmit Status Indicates whether the drop transmit status is enabled. • If this field is 1, the transmit packet status received from MAC is dropped in MTL. • If this field is 0, the transmit packet status received from MAC is forwarded to the application. 0b - Disabled 1b - Enabled
0 —	Reserved

72.18.185 MTL Debug Control (MTL_DBG_CTL)

Offset

Register	Offset
MTL_DBG_CTL	C08h

Function

Controls, along with MTL Debug Status (MTL_DBG_STS), the operation mode of FIFO debug access.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													EIEC		EIEE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STSIE	PKTIE	FIFOSEL		FIFOWREN	FIFORDEN	RSTSEL	RSTALL	Reserved	PKTSTATE		Reserved	BYTEEN		DBGMOD	FDBGEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-17 EIEC	ECC Inject Error Control Indicates the status of the ECC inject error control feature for transmit, receive, and TSO memories. If EIEE = 1, following are the errors inserted based on the value encoded in this field. 00b - 1-bit error 01b - 2-bit errors 10b - 3-bit errors 11b - 1-bit error in the address field
16 EIEE	ECC Inject Error Enable Indicates the status of the ECC error injection feature for transmit, receive, and TSO memories. • If the value if this field is 1, it enables the ECC error injection feature. • If the value if this field is 0, it disables the ECC error injection feature. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
15 STSIE	Transmit Status Available Interrupt Status Enable Indicates whether the transmit packet available interrupt status is enabled. If this field is 1, an interrupt is generated when transmit status is available in Slave mode. 0b - Disabled 1b - Enabled
14 PKTIE	Receive Packet Available Interrupt Status Enable Indicates whether the receive packet available interrupt status is enabled. If this field is 1, an interrupt is generated when EOP of the received packet is written to the receive FIFO. 0b - Disabled 1b - Enabled
13-12 FIFOSEL	FIFO Selected for Access Indicates the FIFO selected for debug access. 00b - Transmit FIFO 01b - Transmit status FIFO (only read access when SLVMOD is set) 10b - TSO FIFO (cannot be accessed when SLVMOD is set) 11b - Receive FIFO
11 FIFOWREN	FIFO Write Enable Indicates the status of FIFO write. If this field is 1, it enables the write operation on the selected FIFO when FIFO debug access is enabled. This field must not be written to 1 when FIFO debug access is disabled, that is, when FDBGEN = 0. Access restrictions apply to this field that clears automatically. Also, writing 0 to the field clears it and writing 1 sets it. 0b - Disabled 1b - Enabled
10 FIFORDEN	FIFO Read Enable Indicates the status of FIFO read. If this field is 1, it enables the read operation on the selected FIFO when FIFO debug access is enabled. This field must not be written to 1 when FIFO debug access is disabled, that is, when FDBGEN = 0. Access restrictions apply to this field that clears automatically. Also, writing 0 to the field clears it and writing 1 sets it. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
9 RSTSEL	Reset Pointers Of Selected FIFO Indicates whether the resetting of pointers for the selected FIFO is enabled. If this field is 1, the pointers of the currently selected FIFO are reset when FIFO debug access is enabled. This field must not be written to 1 when FIFO debug access is disabled, that is, when FDBGEN = 0. Access restrictions apply to this field that clears automatically. Also, writing 0 to the field clears it and writing 1 sets it. 0b - Disabled 1b - Enabled
8 RSTALL	Reset All Pointers Indicates whether the resetting of all the pointers is enabled. If this field is 1, the pointers of all the FIFOs are reset when the FIFO debug access is enabled. This field must not be written to 1 when FIFO debug access is disabled, that is, when FDBGEN = 0. Access restrictions apply to this field that clears automatically. Also, writing 0 to the field clears it and writing 1 sets it. 0b - Disabled 1b - Enabled
7 —	Reserved
6-5 PKTSTATE	Encoded Packet State Is used to write the control information to the transmit FIFO or receive FIFO. These are the values related to the transmit FIFO: • 00: Packet data • 01: Control word • 10: SOP data • 11: EOP data These are the values related to the receive FIFO: • 00: Packet data • 01: Normal status • 10: Last status • 11: EOP 00b - Packet data

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Control word/normal status 10b - SOP data/last status 11b - EOP data/EOP
4 —	Reserved
3-2 BYTEEN	Byte Enables Indicates the number of data bytes valid in the data register during a write operation. This is valid only when MTL_DBG_STS[PKTSTATE] is 2'b10 (EOP), and the transmit FIFO or receive FIFO is selected. 00b - Byte 0 valid 01b - Byte 0 and byte 1 valid 10b - Byte 0, byte 1, and byte 2 valid 11b - All four bytes valid
1 DBGMOD	Debug Mode Access to FIFO Indicates the status of Debug mode access to FIFO. If this field is 1, it indicates that the current access to FIFO is read, write, and debug. In this mode, the following access types are allowed: • Read and write access to transmit FIFO, TSO FIFO, and receive FIFO • Read access to transmit status FIFO If this field is 0, it indicates that the current access to FIFO is slave access bypassing DMA. In this mode, the following access types are allowed: • Write access to transmit FIFO • Read access to receive FIFO and transmit status FIFO 0b - Disabled 1b - Enabled
0 FDBGEN	FIFO Debug Access Enable Indicates the status of FIFO debug access. • If this field is 1, it indicates that the FIFO debug mode access is enabled. • If the value of the field is 0, it indicates that FIFO can be accessed only through a master interface. 0b - Disabled 1b - Enabled

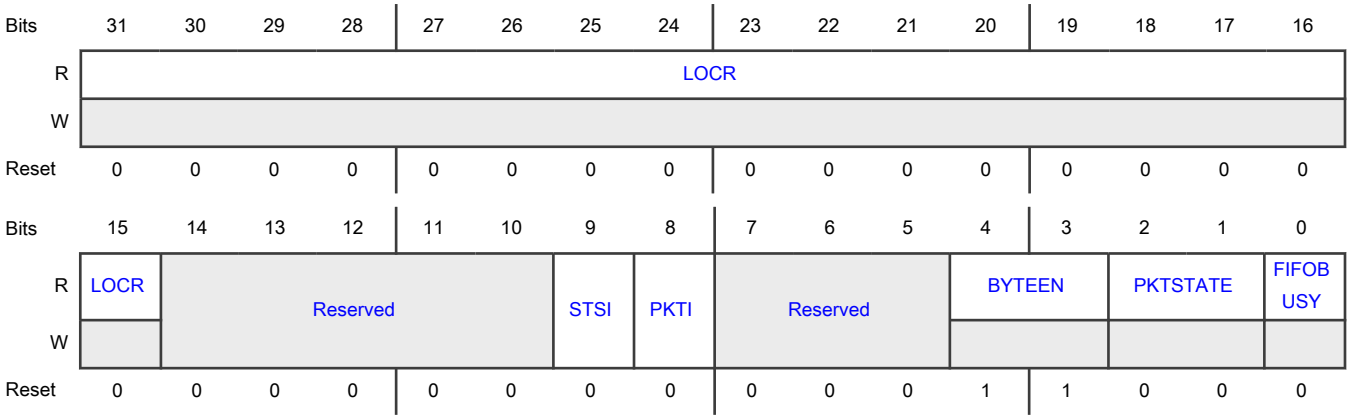
72.18.186 MTL Debug Status (MTL_DBG_STS)

Offset

Register	Offset
MTL_DBG_STS	C0Ch

Function
Contains the status of FIFO debug access.

Diagram



Fields

Field	Function
31-15 LOCR	Remaining Locations In FIFO Indicates the remaining locations in FIFO. For Slave Access mode, the field indicates the space available in the selected FIFO, and for Debug Access mode, this field contains the write or read pointer value of the selected FIFO during the write or read operation, respectively. Reset: In single transmit queue configurations, (DWC_EQOS_TXFIFO_SIZE/(DWC_EQOS_DATAWIDTH/8)), otherwise 0000h.
14-10 —	Reserved
9 STSI	Transmit Status Available Interrupt Status Indicates whether the transmit status available interrupt status is detected. If this field is 1, it indicates that Slave mode transmit packet is transmitted, and the status is available in transmit status FIFO. The field resets when you write 1 to it. 0b - Not detected 1b - Detected
8	Receive Packet Available Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
PKTI	Indicates whether the receive packet available interrupt status is detected. If this field is 1, it indicates that the MAC layer has written the EOP of the received packet to the receive FIFO. The field resets when you write 1 to it. 0b - Not detected 1b - Detected
7-5 —	Reserved
4-3 BYTEEN	Byte Enables Indicates the number of data bytes valid in the data register during a read operation. This is valid only when MTL_DBG_STS[PKTSTATE] is 2'b10 (EOP), and the transmit FIFO or receive FIFO is selected. 00b - Byte 0 valid 01b - Byte 0 and byte 1 valid 10b - Byte 0, byte 1, and byte 2 valid 11b - All four bytes valid
2-1 PKTSTATE	Encoded Packet State Is used to get the control or status information of the selected FIFO. These are the values related to the transmit FIFO. • 00: Packet data • 01: Control word • 10: SOP data • 11: EOP data These are the values related to the receive FIFO. • 00: Packet data • 01: Normal status • 10: Last status • 11: EOP This field is applicable only for transmit and receive FIFOs during a read operation. 00b - Packet data 01b - Control word/normal status 10b - SOP data/last status 11b - EOP data/EOP
0 FIFOBUSY	FIFO Busy Indicates whether the FIFO busy status is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 1, it indicates that a FIFO operation is in progress in MAC and the content of the following is invalid: • All other fields of this register • All the fields of MTL FIFO Debug Data (MTL_FIFO_Debug_Data) 0b - Not detected 1b - Detected

72.18.187 MTL FIFO Debug Data (MTL_FIFO_Debug_Data)

Offset

Register	Offset
MTL_FIFO_Debug_Data	C10h

Function
Contains the data to be written to or read from the FIFOs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	FDBGDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FDBGDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 FDBGDATA	FIFO Debug Data Contains the data to be written to the transmit FIFO, receive FIFO, or TSO FIFO during a debug or slave access write operations. The field contains the data read from the transmit FIFO, receive FIFO, TSO FIFO, or transmit status FIFO during debug or slave access read operations.

72.18.188 MTL Interrupt Status (MTL_Interrupt_Status)

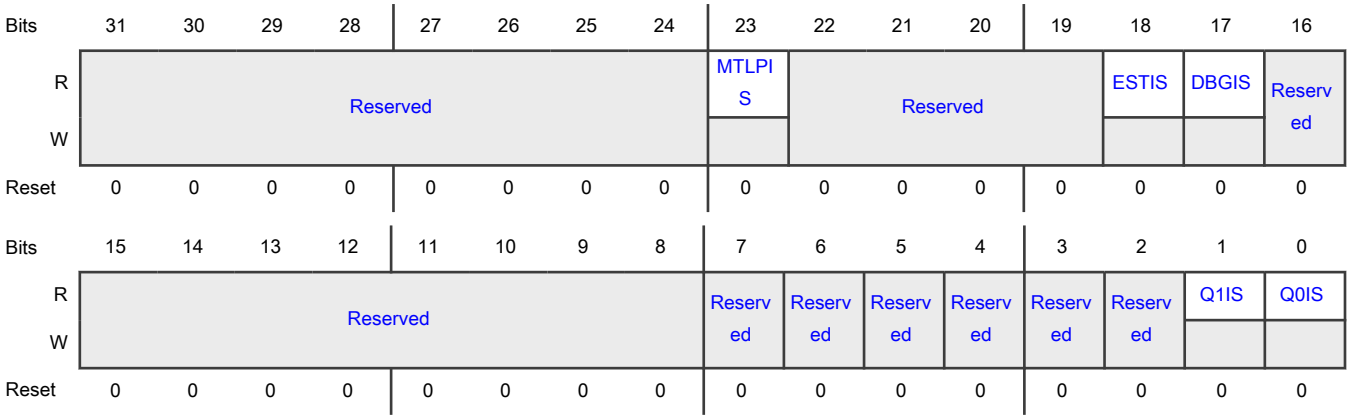
Offset

Register	Offset
MTL_Interrupt_Status	C20h

Function

Determines the interrupt status of MAC and MTL queues. The software driver (application) reads this register during an interrupt service routine or polling to perform this function.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 MTLPIS	MTL Receive Parser Interrupt Status Indicates whether the MTL receive parser interrupt status is detected. This field indicates an interrupt from the receive parser block. To reset this field, the application must read MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status) to get the exact cause of the interrupt and clear its source. 0b - Not detected 1b - Detected
22-19 —	Reserved
18 ESTIS	EST (TAS- 802.1Qbv) Interrupt Status Indicates whether the EST (TAS- 802.1Qbv) interrupt status is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field indicates an interrupt event during the operation of 802.1Qbv. To reset the field, the application must clear the error or event that caused the interrupt. 0b - Not detected 1b - Detected
17 DBGIS	Debug Interrupt Status Indicates whether the debug interrupt status is detected. This field indicates an interrupt event during the slave access. To reset the field, the application must read MTL Debug Status (MTL_DBG_STS) to get the exact cause of the interrupt and clear its source. 0b - Not detected 1b - Detected
16 —	Reserved
15-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 Q1IS	Queue 1 Interrupt Status Indicates whether the queue 1 interrupt status is detected. To reset this field, the application must read MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status) to get the exact cause of the interrupt and clear its source. 0b - Not detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Detected
0 Q0IS	Queue 0 Interrupt Status Indicates whether the queue 0 interrupt status is detected. To reset this field, the application must read MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status) to get the exact cause of the interrupt and clear its source. 0b - Not detected 1b - Detected

72.18.189 MTL Receive Queue DMA Map 0 (MTL_RxQ_DMA_Map0)

Offset

Register	Offset
MTL_RxQ_DMA_Map0	C30h

Function
Reserves in EQOS-CORE and EQOS-MTL configurations.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			Reserved	Reserved			Reserved	Reserved			Reserved	Reserved			Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved			Q1DDMACH	Reserved			Q1MDMACH	Reserved			Q0DDMACH	Reserved			Q0MDMACH
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved.
28 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-25 —	Reserved.
24 —	Reserved.
23-21 —	Reserved.
20 —	Reserved.
19-17 —	Reserved.
16 —	Reserved.
15-13 —	Reserved.
12 Q1DDMACH	Queue 1 Enabled for DA-based DMA Channel Selection Enables or disables queue 1 for DA-based DMA channel selection. If this field is 1, it indicates that the packets received in queue 1 are routed to a particular DMA channel as decided in the MAC receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When this field becomes 0, it indicates that the packets received in queue 1 are routed to the DMA Channel programmed in the MTL_RxQ_DMA_Map0[Q1MDMACH] (bits [10:8]). 0b - Disabled 1b - Enabled
11-9 —	Reserved.
8 Q1MDMACH	Queue 1 Mapped to DMA Channel Controls the routing of the received packet in Queue 1 to the DMA channel: 000b - DMA Channel 0 001b - DMA Channel 1 010b - DMA Channel 2 011b - DMA Channel 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - DMA Channel 4 101b - DMA Channel 5 110b - DMA Channel 6 111b - DMA Channel 7 This field is valid when the MTL_RxQ_DMA_Map0[Q1DDMACH] resets. The field's width depends on the number of RX DMA channels and in some configurations all the values may not be valid . For example, if two RX DMA channels are selected, then only 000b and 001b are valid, the other bits are reserved.
7-5 —	Reserved.
4 Q0DDMACH	Queue 0 Enabled for DA-based DMA Channel Selection Enables or disables queue 0 for DA-based DMA channel selection. If this field is 1, it indicates that the packets received in queue 0 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When this field becomes 0, it indicate that the packets received in queue 0 are routed to the DMA Channel programmed in the MTL_RxQ_DMA_Map0[Q0MDMACH]. 0b - Disable 1b - Enable
3-1 —	Reserved.
0 Q0MDMACH	Queue 0 Mapped to DMA Channel Controls the routing of the packet received in queue 0 to the DMA channel: 000b - DMA channel 0 001b - DMA channel 1 010b - DMA channel 2 011b - DMA channel 3 100b - DMA channel 4 101b - DMA channel 5 110b - DMA channel 6 111b - DMA channel 7 This field is valid when the MTL_RxQ_DMA_Map0[Q0DDMACH] resets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The field's width depends on the number of RX DMA channels in some configurations all the values may not be valid. For example, if the number of RX DMA channels selected is 2, only 000 and 001 bits are valid, the other bits are reserved.

72.18.190 MTL TBS Control (MTL_TBS_CTRL)

Offset

Register	Offset
MTL_TBS_CTRL	C40h

Function

Controls the operation of time based scheduling.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LEOS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LEOS								Reserved	LEGOS			Reserved		LEOV	ESTM
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 LEOS	Launch Expiry Offset Computes the launch expiry time. To compute the launch expiry time the value in units of 256 nanoseconds is added to the launch time. The field value is valid only if MTL_TBS_CTRL[LEOV] is 1. Max value: 999,999,999 nanosecond should be smaller than CTR-1 value when ESTM mode = 1, because this value is a modulo CTR value.
7 —	Reserved.
6-4	Launch Expiry GSN Offset

Table continues on the next page...

Table continued from the previous page...

Field	Function
LEGOS	Computes the Launch expiry time. To compute the launch expiry time the number of GSN slots is to be added to the launch GSN. The field value is valid only if MTL_TBS_CTRL[LEOV] is 1.
3-2 —	Reserved.
1 LEOV	Launch Expiry Offset Valid Indicates if MTL_TBS_CTRL[LEOS] is valid or invalid. When this field is 1, it indicates that MTL_TBS_CTRL[LEOS] is valid. When not 1, it indicates that the launch expiry Offset is not valid and the MTL must not check for launch expiry time. 0b - Invalid 1b - Valid
0 ESTM	EST offset Mode Enables or disables EST Offset mode. If this field is 1, the launch time value used in Time Based Scheduling is interpreted as an EST offset value and is added to the BTR of the current list. When this field becomes 0, the launch time value is used as an absolute value that must be compared with the system time [39:8]. 0b - Disabled 1b - Enabled

72.18.191 MTL EST Control (MTL_EST_Control)

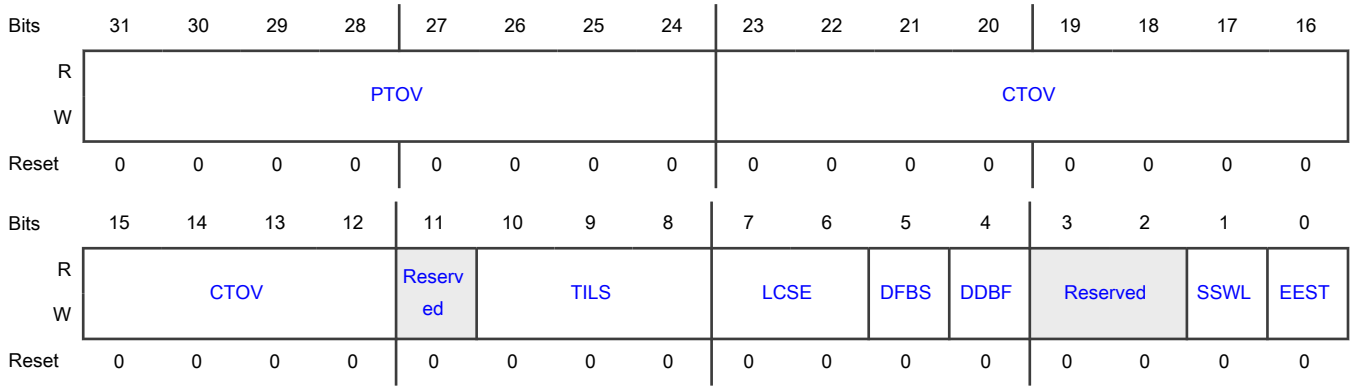
Offset

Register	Offset
MTL_EST_Control	C50h

Function

Controls the operation of enhancements to scheduled transmission (IEEE802.1Qbv).

Diagram



Fields

Field	Function
31-24 PTOV	PTP Time Offset Value Indicates the PTP time offset value. You must multiply 6 to the value of PTP clock period in nanoseconds. At the beginning of the installation of a new GCL this value avoid transmission overruns.
23-12 CTOV	Current Time Offset Value Provides a 12 bit time offset value in nano second which is added to the current time to compensate for all the implementation pipeline delays such as the CDC sync delay, buffering delays, data path delays and so on. This offset ensures that the impact of gate controls is visible on the line exactly at the pre-determined schedule (or as close to the schedule as possible).
11 —	Reserved.
10-8 TILS	Time Interval Left Shift Amount Provides the left shift amount for the programmed time interval values used in the gate control lists. 000b - No left shift needed (equal to x1ns) 001b - Left shift TI by 1 bit (equal to x2ns) 010b - Left shift TI by 2 bits (equal to x4ns) - - 100b - Left shift TI by 7 bits (equal to x128ns) On the basis of configuration you must consider one or more bits of this field as reserved or read-only.
7-6 LCSE	Loop Count to Report Scheduling Error Provides programmable number of GCL list iterations before reporting an HLBS error which is defined in MTL_EST_Status.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - 4 iterations 01b - 8 iterations 10b - 16 iterations 11b - 32 iterations
5 DFBS	Drop Frames Causing Scheduling Error Provides the status of the frames causing scheduling error. If this field is 1 then the frames report to cause an HOL blocking because the MTL_EST_Status[HLBS] is not scheduled after 4,8,16,and 32 (on the basis of MTL_EST_Control[LCSE]) GCL iterations are dropped. 0b - Do not drop frames 1b - Drop frames
4 DDBF	Do not Drop Frames during Frame Size Error Provides status of frames during Frame Size Error. If this field is 1, it indicates that the frames are not dropped during head-of-line blocking because of the frame size error (MTL_EST_Status[HLBF]). 0b - Drop frames during frame size error 1b - Do not Drop frames during frame size error
3-2 —	Reserved.
1 SSWL	Switch to Software Owned List If this field is 1, it indicates that the software has programmed that list that it currently owns (SWOL) and the hardware should switch to the new list based on the new BTR. Hardware clears this bit when the switch to the SWOL happens to indicate the completion of the switch or when an BTR error (BTRE in Status register) is set. When BTRE is set this bit is cleared but SWOL is not updated as the switch was not successful. Access restriction apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. Enables or disables the switch to the software owned list. 0b - Disabled 1b - Enabled
0 EEST	Enable EST Enables or disables EST. If the field becomes 0, it indicate that the gate control list processing is halted and all gates are assumed to be in open state. The field must be 1 for the hardware to start processing the gate control lists. During the toggle from 0 to 1, when the MTL_EST_Control[SSWL] is 1, the gate control list processing starts. During the configuration when DWC_EQOS_ASP_ECC is selected, the hardware resets this field and disables the EST function if any uncorrectable error is detected in the EST memory .

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled

72.18.192 MTL EST Status (MTL_EST_Status)

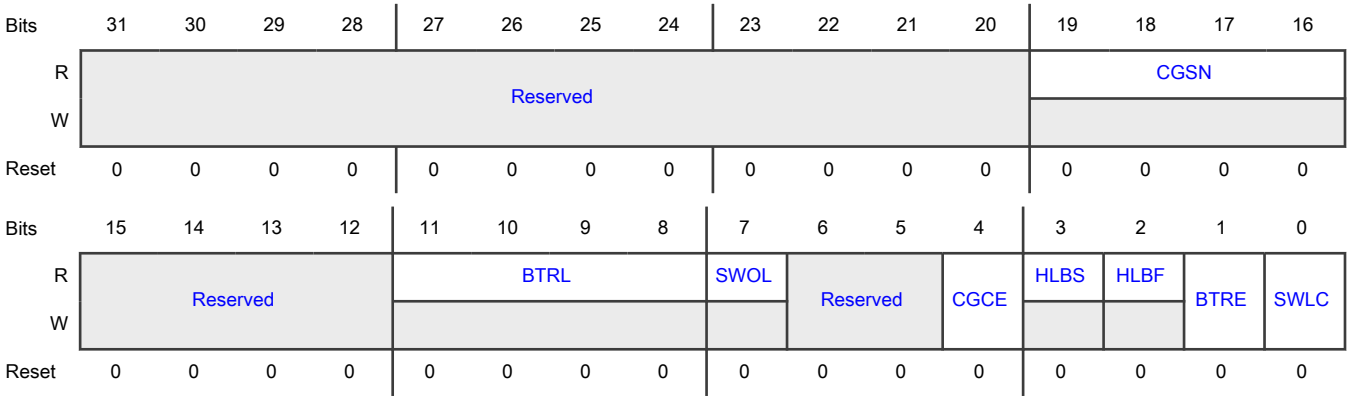
Offset

Register	Offset
MTL_EST_Status	C58h

Function

Provides status of enhancements to scheduled transmission (IEEE802.1Qbv).

Diagram



Fields

Field	Function
31-20 —	Reserved.
19-16 CGSN	Current GCL Slot Number Indicates the slot number of the GCL list. Slot number is a modulo 16 count of the GCL List loops executed so far. Even if a new GCL list is installed, the count is incremental.
15-12 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-8 BTRL	BTR Error Loop Count Provides the minimum count (N) for which the equation Current Time =< New BTR + (N * New Cycle Time) becomes true. N = "1111" indicates the iterations exceeded the value of 8 and the hardware was not able to update New BTR to be equal to or greater than Current Time. Software intervention is needed to update the New BTR. Value cleared when BTRE field of this register is cleared.
7 SWOL	S/W owned list When '0' indicates Gate control list number "0" is owned by software and when "1" indicates the Gate Control list "1" is owned by the software. Any reads/writes by the software (using indirect access via GCL_Control) is directed to the list indicated by this value by default. The inverse of this value is treated as HWOL. R/W operations performed by hardware are directed to the list pointed by HWOL by default. 0b - Gate control list number "0" is owned by software 1b - Gate control list number "1" is owned by software
6-5 —	Reserved.
4 CGCE	Constant Gate Control Error This error occurs when the list length (LLR) is 1 and the Cycle Time (CTR) is less than or equal to the programmed Time Interval (TI) value after the optional Left Shifting. The above programming implies Gates are either always Closed or always Open based on the Gate Control values; the same effect can be achieved by other simpler (non TSN) programming mechanisms. Since the implementation does not support such a programming an error is reported. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0b - Constant Gate Control Error not detected 1b - Constant Gate Control Error detected
3 HLBS	Head-Of-Line Blocking due to Scheduling Set when the frame is not able to win arbitration and get scheduled even after 4 iterations of the GCL. Indicates to software a potential programming error. The one hot encoded values of the Queue Numbers that are not able to make progress are indicated in the MTL_EST_Sch_Error register. Bit cleared when MTL_EST_Sch_Error register is all zeros. 0b - Head-Of-Line Blocking due to Scheduling not detected 1b - Head-Of-Line Blocking due to Scheduling detected
2 HLBF	Head-Of-Line Blocking due to Frame Size Set when HOL Blocking is noticed on one or more Queues as a result of none of the Time Intervals of gate open in the GCL being greater than or equal to the duration needed for frame size (or frame fragment size when preemption is enabled) transmission. The one hot encoded Queue numbers that are experiencing HLBF are indicated in the MTL_EST_Frm_Size_Error register. Additionally, the first Queue number that experienced HLBF along with the frame size is captured in MTL_EST_Frm_Size_Capture register. Bit cleared when MTL_EST_Frame_Size_ Error register is all zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Head-Of-Line Blocking due to Frame Size not detected 1b - Head-Of-Line Blocking due to Frame Size detected
1 BTRE	BTR Error Indicates whether the BTR error is detected. When this field is 1, it indicates a programming error in the BTR of SWOL where the programmed value is less than the current time. If the BTRL = "1111", SWOL is not updated and software must reprogram the BTR to a value greater than current time and then set SSWL to reinitiate the switch to SWOL. Else if the value of BTRL < "1111", SWOL is updated and this field indicates the number of iterations (of + CycleTime) taken by hardware to update the BTR to a value greater than Current Time. Access restrictions apply to this field. It sets automatically to 1 on an internal event occurence. Writing 1 clears this field and writing 0 has no effect. 0b - BTR Error not detected 1b - BTR Error detected
0 SWLC	Switch to Software Owned List Complete Indicates whether the switch to software owned list complete is detected. When the field is 1, it indicates that the hardware has successfully switched to the SWOL, and updated the MTL_EST_Status[SWOL] to that effect. This field is 0 when MTL_EST_Control[SSWL] transitions from 0 to 1, or on a software write. Access restrictions apply to this field. It clears automatically and writing 0 to it has no effect. 0b - Not detected 1b - Detected

72.18.193 MTL EST Scheduling Error (MTL_EST_Sch_Error)

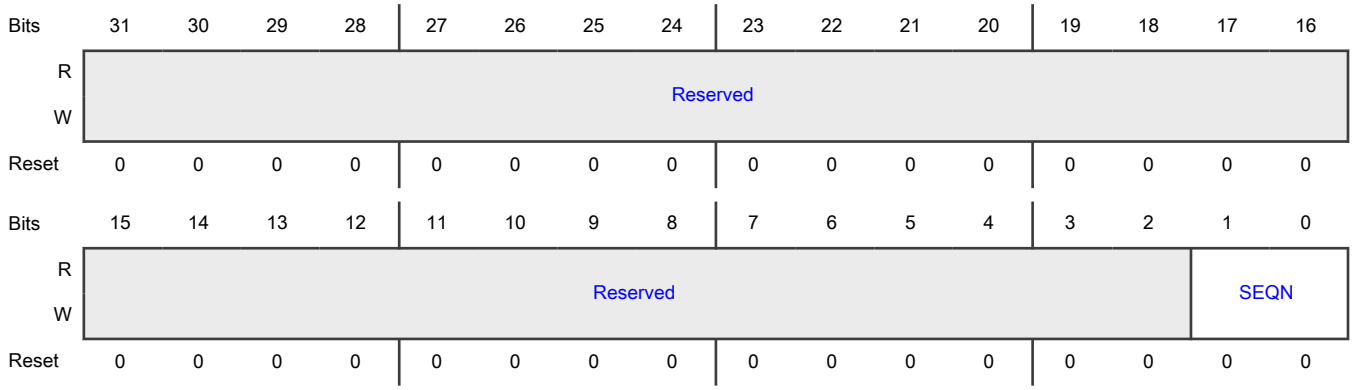
Offset

Register	Offset
MTL_EST_Sch_Error	C60h

Function

Provides the one hot encoded queue numbers that have the scheduling related error (timeout).

Diagram



Fields

Field	Function
31-2 —	Reserved.
1-0 SEQN	Schedule Error Queue Number Provides the one hot encoded queue numbers that have experienced error or timeout described in MTL_EST_Status[HLBS]. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 to it has no effect.

72.18.194 MTL EST Frame Size Error (MTL_EST_Frm_Size_Error)

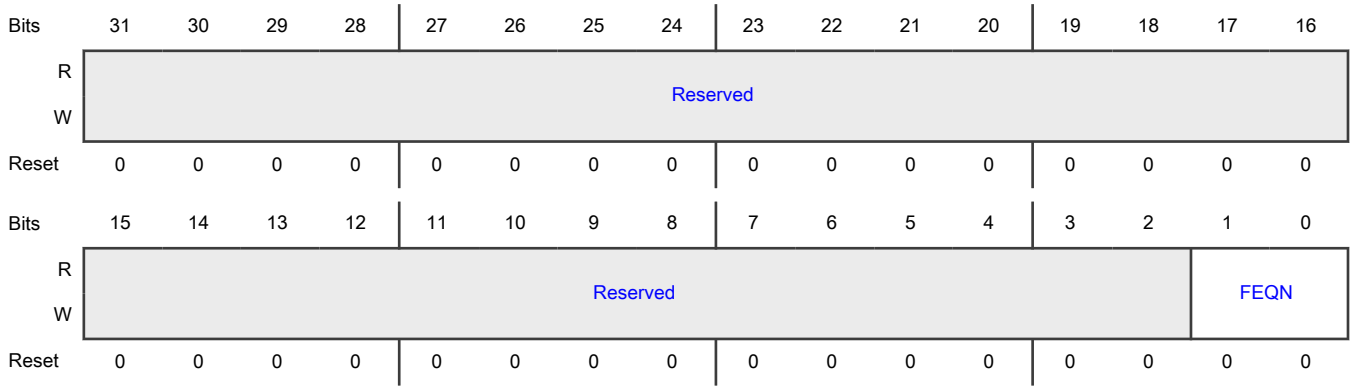
Offset

Register	Offset
MTL_EST_Frm_Size_Error	C64h

Function

Provides the one hot encoded queue numbers that have the frame Size related error.

Diagram



Fields

Field	Function
31-2 —	Reserved.
1-0 FEQN	Frame Size Error Queue Number Provides the one hot encoded queue Nnumbers that have experienced error described in MTL_EST_Status[HLBF]. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 to it has no effect.

72.18.195 MTL EST Frame Size Capture (MTL_EST_Frm_Size_Capture)

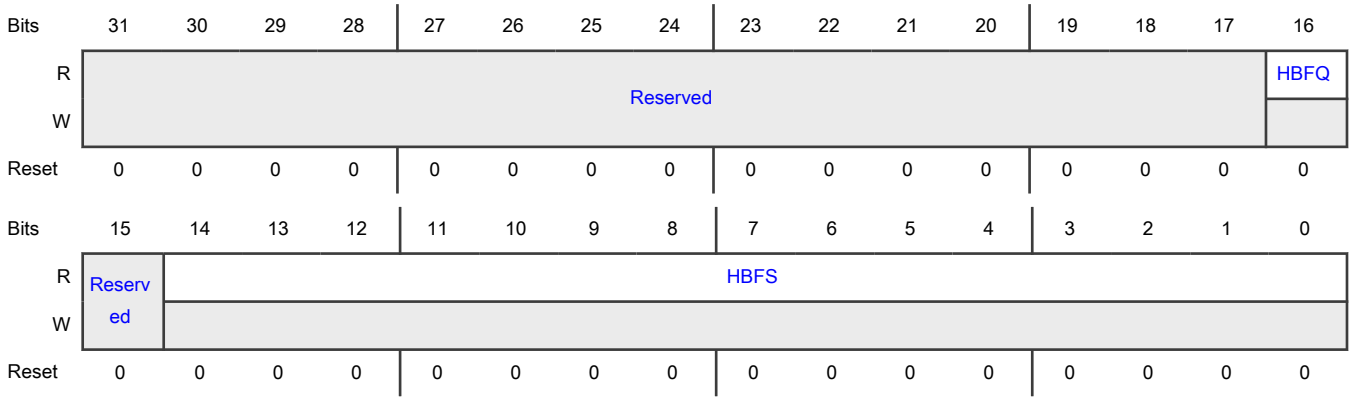
Offset

Register	Offset
MTL_EST_Frm_Size_Capture	C68h

Function

Captures the frame Size and queue number of the first occurrence of the frame Size related error. When you write 0, it captures the data of immediate next occurrence of a similar error.

Diagram



Fields

Field	Function
31-17 —	Reserved.
16 HBFQ	Queue Number of HLBF Captures the binary value of the first queue (number) which experiences the HLBF error (see MTL_EST_Status[HLBF]). Any subsequent queue errors similar to HLBF errors does not alter the value after it is written. After this field is 0, it captures the queue number of the next occurring HLBF error. Field's width is based on the number of transit queues configured; remaining fields are read-only. This field clears when MTL_EST_Frm_Size_Error is all zeros.
15 —	Reserved.
14-0 HBFS	Frame Size of HLBF Captures the frame Size of the dropped frame related to queue number indicated in MTL_EST_Frm_Size_Capture[HBFQ]. If this field is zero, then this register must be considered invalid. This field clears when MTL_EST_Frm_Size_Error is all zeros.

72.18.196 MTL EST Interrupt Enable (MTL_EST_Intr_Enable)

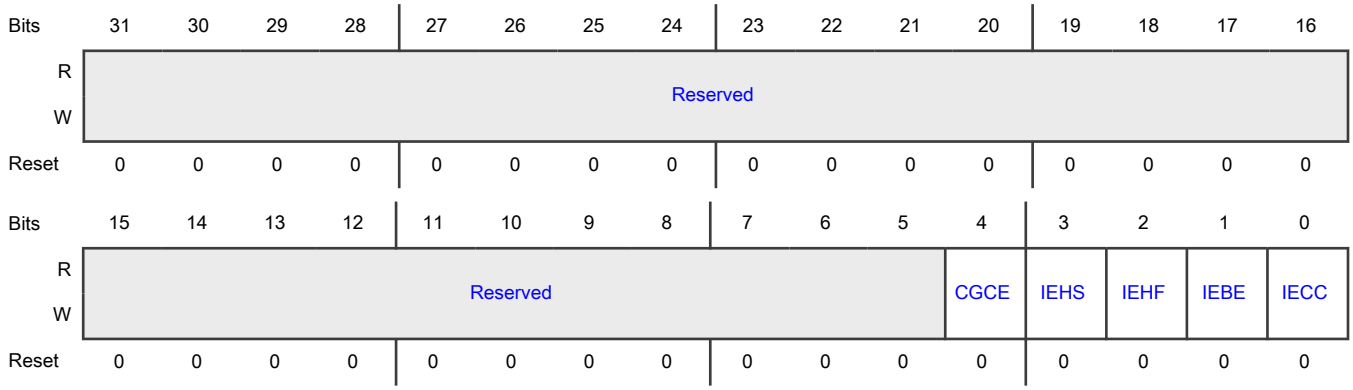
Offset

Register	Offset
MTL_EST_Intr_Enable	C70h

Function

Implements the interrupt enable fields for the various events that generate an interrupt. Bit positions have a 1 to 1 correlation with the status bit positions in MTL_ETS_Status register.

Diagram



Fields

Field	Function
31-5 —	Reserved.
4 CGCE	Interrupt Enable for CGCE Indicates whether an interrupt for MTL_EST_Intr_Enable[CGCE] is enabled. When this field is 1, it generates an interrupt when the constant gate control error occurs and is indicated in the status. When this field becomes 0, this event does not generate an interrupt 0b - Disabled 1b - Enabled
3 IEHS	Interrupt Enable for HLBS Indicates whether an interrupt for MTL_EST_Status[HLBS] is enabled. When this field is 1, it generates an interrupt when the head-of-line blocking due to scheduling issue occurs and is indicated in the status. When this field becomes 0, this event does not generate an interrupt. 0b - Disabled 1b - Enabled
2 IEHF	Interrupt Enable for HLBF Indicates whether an interrupt for MTL_EST_Status[HLBF] is enabled. When this field is 1, it generates an interrupt when the head-of-line blocking due to frame size error occurs and is indicated in the status. When this field becomes 0, this event does not generate an interrupt. 0b - Disabled 1b - Enabled
1 IEBE	Interrupt Enable for BTR Error Indicates whether an interrupt for BTR error is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1, it generates an interrupt when the BTR error occurs and is indicated in the status. When this field becomes 0, this event does not generate an interrupt. 0b - Disabled 1b - Enabled
0 IECC	Interrupt Enable for Switch List Indicates whether the interrupt for switch list is enabled. When this field is 1, it generates an interrupt when the configuration change is successful and the hardware switches to the new list. When this field becomes 0, this event does not generate an interrupt. 0b - Disabled 1b - Enabled

72.18.197 MTL EST GCL Control (MTL_EST_GCL_Control)

Offset

Register	Offset
MTL_EST_GCL_Control	C80h

Function

Provides the control information for reading and writing to the gate control lists.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								ESTEIEC		ESTEIEE	ERR0	Reserved			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR								Reserved		DBGB	DBGM	Reserved	GCRR	R1W0	SRWO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23-22 ESTEIEC	ECC Inject Error Control for EST Memory Provides the status of inject error control for EST memory. When MTL_EST_GCL_Control[ESTEIEE] = 1, it indicates that on the basis of the value encoded in this field following errors are inserted. This field is valid only if you select DWC_EQOS_ASP_ECC feature during the configuration, otherwise it is reserved. 00b - Insert 1 bit error 01b - Insert 2 bit errors 10b - Insert 3 bit errors 11b - Insert 1 bit error in address field
21 ESTEIEE	EST ECC Inject Error Enable Indicates whether the EST ECC inject error is enabled. If this field along with MTL_EST_Control[EEST] = 1, it enables the ECC error injection feature. When this field becomes 0, it disables the ECC error injection feature. 0b - Disabled 1b - Enabled
20 ERR0	If this field is 1, it indicates that when the software writes to GCL the last write operation was aborted and when MTL_EST_Control[SSWL] is 1, GCL registers are prohibited. Access restriction apply to this field and automatically becomes 1 on an internal event occurrence. Writing 1 to this field clears it and writing 0 to this field has no effect. 0b - ERR0 is disabled 1b - ERR1 is enabled
19-16 —	Reserved.
15-8 ADDR	Gate Control List Address: (GCLA when GCRR is "0"). Provides the address (row number) of the gate control list at which the R/W operation has to be performed. By default the gate control list to which MTL_EST_Status[SWOL] points is selected for read and write. If MTL_EST_GCL_Control[DBGM] is 1, it indicates that MTL_EST_GCL_Control[DBGB] gives Debug mode access to read and write. The field's width depends on DWC_EQOS_EST_DEP and the unused bits must be treated as read only. Gate Control list Related Registers Address: (GCRA when GCRR is "1"). By default the GCL related register set pointed by MTL_EST_Status[SWOL] is selected for read or write. If MTL_EST_GCL_Control[DBGM], it indicates that MTL_EST_GCL_Control[DBGB] gives Debug mode access to read write. Lower 3 bits are only used in this mode, higher order bits are treated as dont cares.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - BTR Low (31:0) 001b - BTR High (63:31) 010b - CTR Low (31:0) 011b - CTR High (39:32) 100b - TER (31:0) 101b - LLR (n:0) (where n is (log{DWC_EQOS_EST_DEP} / log2)) Others - Reserved
7-6 —	Reserved.
5 DBGB	Debug Mode Bank Select Indicates whether the read and write in Debug mode is directed to bank 0 or bank 1. When this field is 0, it indicates that read or write in Debug mode must be directed to bank 0 (GCL0 and corresponding Time related registers). When this field is 1, it indicates that read or write in Debug mode should be directed to bank 1 (GCL1 and corresponding Time related registers). When MTL_EST_GCL_Control[DBGM] is 1 you can use this value and overrides with the MTL_EST_Status[SWOL] value. 0b - Directed to bank 0 1b - Directed to bank 1
4 DBGM	Debug Mode Enables or disables Debug mode. When this field is 1, it indicates that the read and write is in Debug mode where the MTL_EST_GCL_Control[DBGB] value explicitly provides the memory bank (for GCL and time related registers). When this field is 0, it indicates that MTL_EST_Status[SWOL] determines which bank to use. 0b - Disabled 1b - Enabled
3 —	Reserved.
2 GCRR	Gate Control Related Registers Enables or disables gate control related registers. When this field is 1, it indicates that the read or write access is for the GCL related registers (BTR, CTR, TER, LLR) for which GCRA provides the address. When this field is 0, it indicates that read and write must be directed to GCL from the address which GCLA provides. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 R1W0	Read '1', Write '0' Indicates whether this field performs read operation or write operation. When this field is 1 it indicates that it performs a read operation. When this field is 0 it indicates that it performs a write operation. 0b - Write operation 1b - Read operation
0 SRWO	Start Read/Write Operation Indicates whether the read/write operation is enabled. When this field is 1, it indicates that a read/write operation has started and is in progress. When hardware resets this field and MTL_EST_GCL_Control[ERR0] = 1 , it indicates that the read/write operation has completed or an error has occurred. Reads: When this field becomes 0, it indicates that MTL_EST_GCL_Data reads data. Writes: Before writing 1 to this field, you must program MTL_EST_GCL_Data with write data. Access restrictions apply to this field. It clears automatically. Writing 1 sets this field and writing 0 to it has no effect. 0b - Disabled 1b - Enabled

72.18.198 MTL EST GCL Data (MTL_EST_GCL_Data)

Offset

Register	Offset
MTL_EST_GCL_Data	C84h

Function

Holds the read data or write data in case of reads and writes respectively.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GCD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GCD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GCD	Gate Control Data Provides the data corresponding to the address selected in the MTL_EST_GCL_Control. This field is used for both read and write operations.

72.18.199 MTL FPE Control Status (MTL_FPE_CTRL_STS)

Offset

Register	Offset
MTL_FPE_CTRL_STS	C90h

Function

Controls the operation of, and provides status for frame preemption (IEEE802.1Qbu/802.3br).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			HRS	Reserved											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						PEC		Reserved						AFSZ	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved.
28 HRS	Hold/Release Status Indicates whether a set-and-release-MAC operation which was last executed and the pMAC is in release state or in hold state. 0b - Release state 1b - Hold state
27-16 —	Reserved.
15-10 —	Reserved.
9-8 PEC	Preemption Classification When this field is 1, it indicates that you must classify the corresponding queue as preemptable. When this field is 0, it indicates that you must classify the queue as express. When set indicates the corresponding Queue must be classified as preemptable, when '0' Queue is classified as express. When both EST (Qbv) and Preemption are enabled, Queue-0 is always assumed to be preemptable. When you enables EST (Qbv), the preemptable queues are always assumed to be in open state in gate control list.
7-2 —	Reserved.
1-0 AFSZ	Additional Fragment Size Indicates that, in units of 64 bytes, the minimum number of bytes over 64 bytes required in non-final fragments of preempted frames. The minimum non-final fragment size is (AFSZ +1) * 64 bytes.

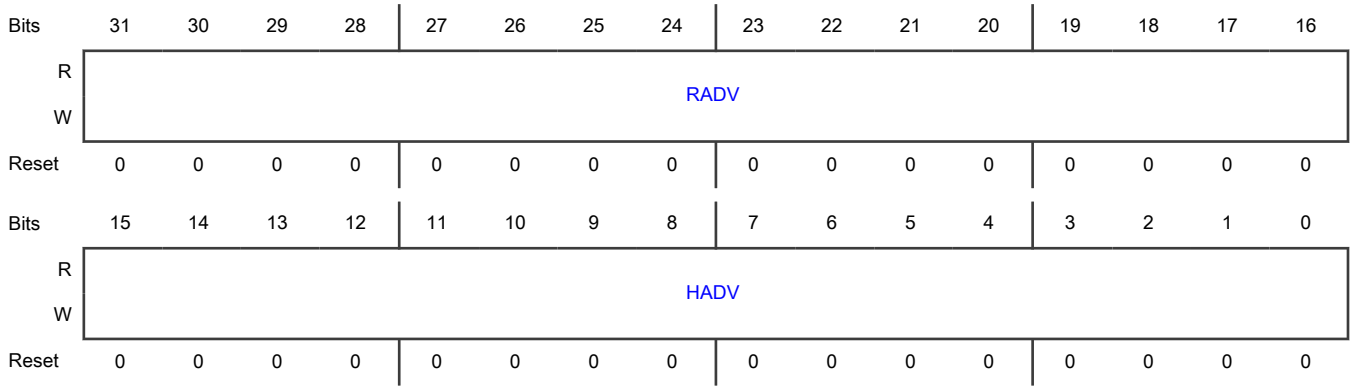
72.18.200 MTL FPE Advance (MTL_FPE_Advance)

Offset

Register	Offset
MTL_FPE_Advance	C94h

Function
Holds the hold and release advance time.

Diagram



Fields

Field	Function
31-16 RADV	Release Advance Provides the maximum time in nanoseconds that can elapse between issuing a release to the MAC and the MAC being ready to resume transmission of preemptable frames, in the absence of there being any express frames available for transmission.
15-0 HADV	Hold Advance Provides the maximum time in nanoseconds that can elapse between issuing a hold to the MAC and the MAC ceasing to transmit any preemptable frame that is in the process of transmission or any preemptable frames that are queued for transmission.

72.18.201 MTL Rx Parser Control Status (MTL_RXP_Control_Status)

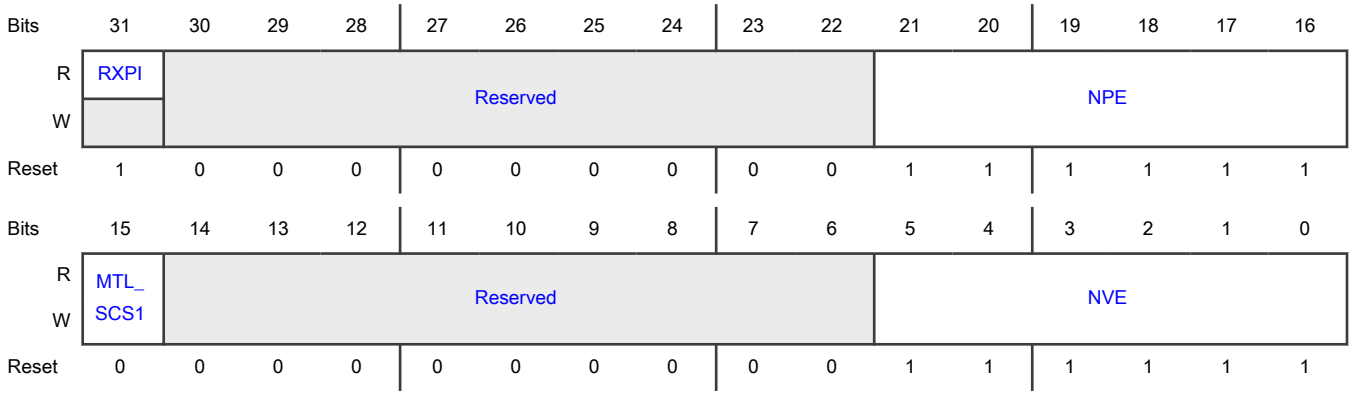
Offset

Register	Offset
MTL_RXP_Control_Status	CA0h

Function

Establishes the operating mode of receive parser and provides some status.

Diagram



Fields

Field	Function
31 RXPI	RX Parser in Idle State Indicates whether the receive parser is in idle state. When this field is 1, it indicates that the receive parser is in idle state and waiting for the processing of a new packet. When parser disables, you can use this field as a handshake with software and also the software can update the receive parser instruction table when you write 1 to this field. 0b - Not in Idle state 1b - Idle state
30-22 —	Reserved.
21-16 NPE	Number of parsable entries in the Instruction table Indicates the number of parsable entries in the instruction memory. This is used in receive parser logic to detect programming error. MTL_RXP_Interrupt_Control_Status[NPEOVIS] = 1, if the number of parsed entries for a packet is more than this entry.
15 MTL_SCS1	MTL_SCS1 Is reserved for NXP internal use. The value of this field must always be 0 unless NXP instructs you otherwise. Writing 1 to any of the bits of this field might cause unexpected behavior in th IP.
14-6 —	Reserved.
5-0 NVE	Number Of Valid Entry Address Or Index In The Instruction Table Indicates the number of valid entry address or index in the instruction memory.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The maximum valid entry address is = NVE + 1 and the addresses or indices range from 0 to 32. Therefore, if the value of this field is 32, the number of valid entry address is 33. This value is used in receive parser logic to detect if any programming errors exist. When parsing, the module writes 1 to NVEOVIS if the number of the memory address is found to be more than the defined maximum valid entry address number. **NOTE** The minimum value of this field must be two.

72.18.202 MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status)

Offset

Register	Offset
MTL_RXP_Interrupt_Control_Status	CA4h

Function

Provides enable control for the interrupts and also provides interrupt status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved												PDRFIE	FOOVIE	NPEOVIE	NVEOVIE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												PDRFIS	FOOVIS	NPEOVIS	NVEOVIS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved.
19 PDRFIE	Packet Drop due to RF Interrupt Enable Indicates whether the packet drop due to RF interrupt is enabled. If this field is 1, it enables the PDRFIS interrupt. When this field becomes 0, it disables the PDRFIS interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
18 FOOVIE	Frame Offset Overflow Interrupt Enable Indicates whether the frame offset overflow interrupt is enabled. If this field is 1, it enables the FOOVIS interrupt. When this field becomes 0, it disables the FOOVIS interrupt. 0b - Disabled 1b - Enabled
17 NPEOVIE	Number of Parsable Entries Overflow Interrupt Enable Enables or disables the number of parsable entries overflow interrupt. If this field is 1, it enables the NPEOVIS interrupt. When this field becomes 0, it disables the NPEOVIS interrupt. 0b - Disable 1b - Enable
16 NVEOVIE	Number of Valid Entries Overflow Interrupt Enable Enables or disables the number of valid entries overflow interrupt. If this field is 1, it enables the NVEOVIS interrupt. When this field becomes 0, it disables the NVEOVIS interrupt. 0b - Disable 1b - Enable
15-4 —	Reserved.
3 PDRFIS	Packet Dropped due to RF Interrupt Status Indicates whether the packet dropped due to RF interrupt status is detected. If the Rx Parser result says to drop the packet by setting RF=1 in the instruction memory, then this bit is set to 1. This field clears when the application writes 1 to this field. Access apply to this field. It automatically sets to 1 on an internal event occurrence. Writing 1 clears this field and writing 0 to it has no effect. 0b - Not detected 1b - Detected
2	Frame Offset Overflow Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
FOOVIS	Indicates whether the frame offset overflow interrupt status is detected. If this field is 1, it indicates that when parsing, the instruction table entry's frame offset is more than the EOF offset. This field clears when the application writes 1 to this field. Access apply to this field. It automatically sets to 1 on an internal event occurrence. Writing 1 clears this field and writing 0 to it has no effect. 0b - Not detected 1b - Detected
1 NPEOVIS	Number of Parsable Entries Overflow Interrupt Status Indicates whether the number of parsable entries overflow interrupt status is detected. If this field is 1, it indicates that when parsing a packet the number of parsed entries are more than NPE[] (MTL_RXP_Control_Status[NPE]). This field clears when the application writes 1 to this field. Access apply to this field. It automatically sets to 1 on an internal event occurrence. Writing 1 clears this field and writing 0 to it has no effect. 0b - Not detected 1b - Detected
0 NVEOVIS	Number of Valid Entry Address/Index Overflow Interrupt Status Indicates whether the number of valid entry address or index overflow interrupt status is detected. If this field is 1, it indicates that when parsing, the instruction address is more than MTL_RXP_Control_Status[NVE]. For example, when MTL_RXP_Control_Status[NVE] = 31, the maximum valid entry address/index is NVE+1 that is 32 (addresses/indices = 0 to 32, or 33 entries). MTL_RXP_Interrupt_Control_Status[NVEOVIS] = 1 when currently processed entry indicates that the next address is 33 or more, that is 34th or later entries. This field clears when the application writes 1 to this field. Access apply to this field. It automatically sets to 1 on an internal event occurrence. Writing 1 clears this field and writing 0 to it has no effect. 0b - Not detected 1b - Detected

72.18.203 MTL Rx Parser Drop Count (MTL_RXP_Drop_Cnt)

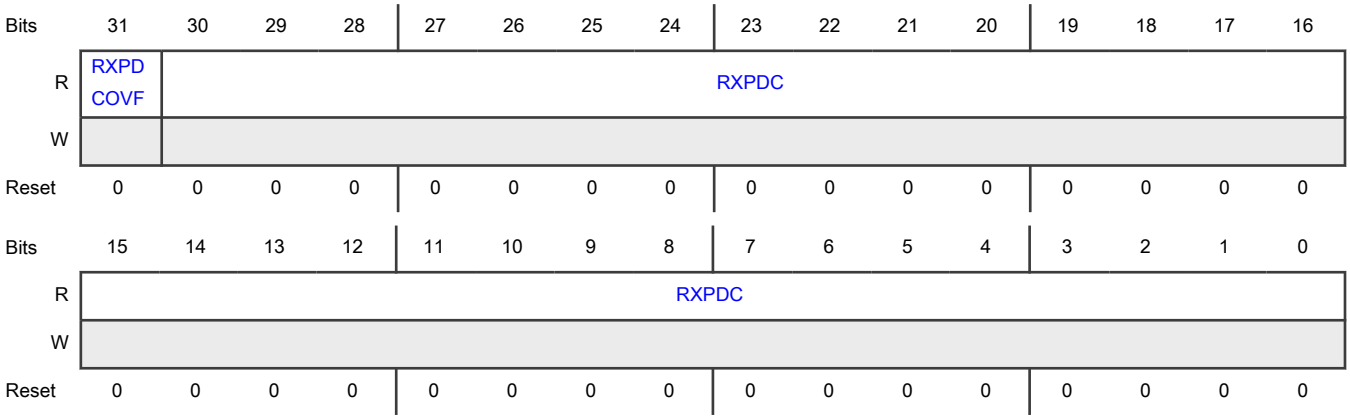
Offset

Register	Offset
MTL_RXP_Drop_Cnt	CA8h

Function

Provides the drop count of receive parser initiated drops.

Diagram



Fields

Field	Function
31 RXPDCOVF	Rx Parser Drop Counter Overflow Bit Indicates whether the receive parser drop count overflow has occurred. When this field is 1, it indicates that the MTL_RXP_Drop_Cnt[RXPDC] counter field has crossed the maximum limit. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not occurred 1b - Occurred
30-0 RXPDC	Rx Parser Drop Count Provides the receive parser drop count. This 31-bit counter is implemented whenever a receive parser drops a packet due to RF = 1. The counter clears when you read the register.

72.18.204 MTL Rx Parser Error Count (MTL_RXP_Error_Cnt)

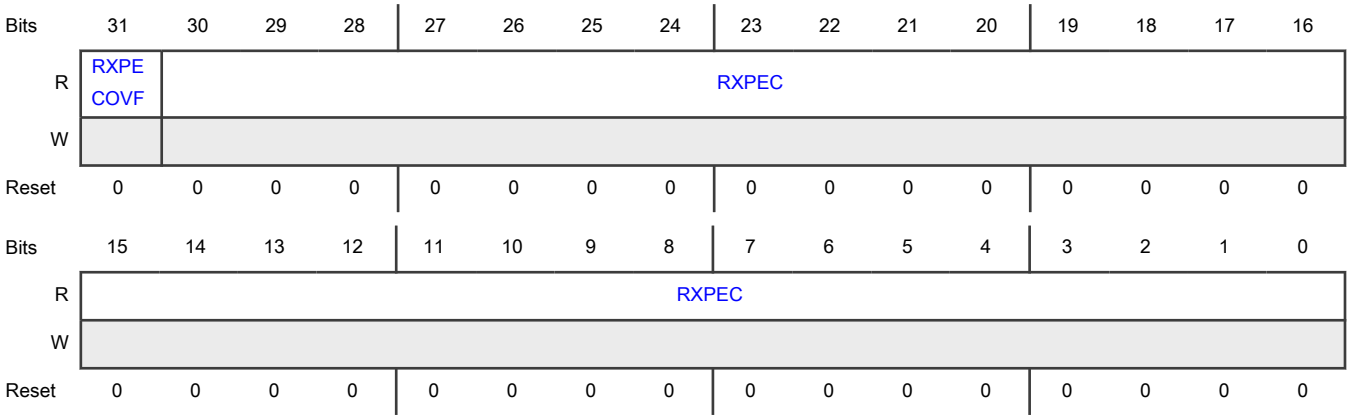
Offset

Register	Offset
MTL_RXP_Error_Cnt	CACh

Function

Provides the receive parser related error occurrence count.

Diagram



Fields

Field	Function
31 RXPECOVF	Rx Parser Error Counter Overflow Bit Indicates whether the receive parser error counter overflow has occurred. When this field is 1, it indicates that MTL_RXP_Error_Cnt[RXPEC] counter field has crossed the maximum limit. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not occurred 1b - Occurred
30-0 RXPEC	Rx Parser Error Count Provides the receive parser error count. This 31-bit counter is implemented whenever a receive parser encounters following error scenarios: Entry address >= NVE[] Number parsed entries >= NPE[] Entry address > EOF data entry address The counter clears when you read the register.

72.18.205 MTL Rx Parser Indirect Access Control Status (MTL_RXP_Indirect_Acc_Control_Status)

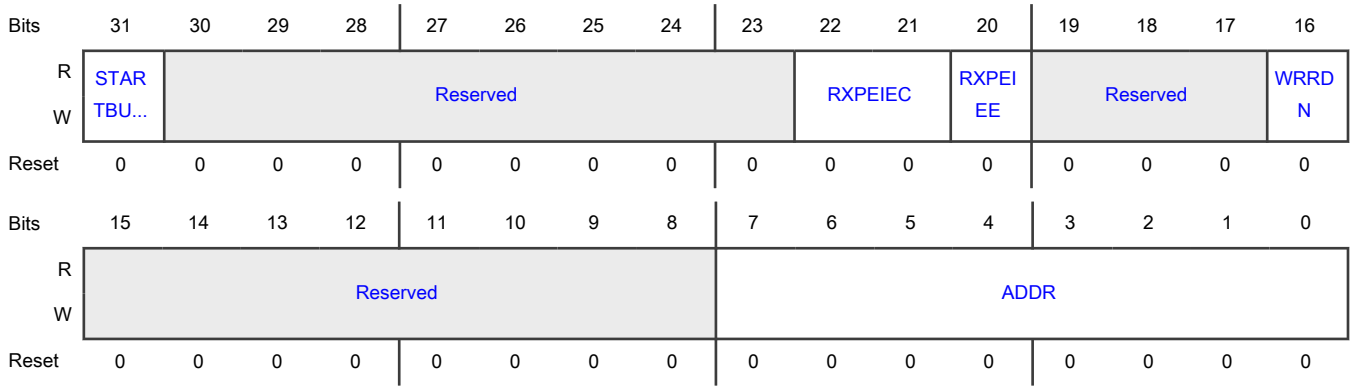
Offset

Register	Offset
MTL_RXP_Indirect_Acc_Control_Status	CB0h

Function

Provides the indirect access control and status for receive parser memory.

Diagram



Fields

Field	Function
31 STARTBUSY	FRP Instruction Table Access Busy When this bit is set to 1 by the software then it indicates to start the Read/Write operation from/to the Rx Parser Memory. Software should read this bit as 0 before issuing read or write request to access the Parser Memory Instructions. This bit when set to 1 indicates that hardware is busy until its gets cleared by hardware and software should not issue any read or write operation. 0b - hardware not busy 1b - hardware is busy (Read/Write operation from/to the Rx Parser Memory)
30-23 —	Reserved.
22-21 RXPEIEC	ECC Inject Error Control for Rx Parser Memory Controls the ECC inject error for receive parser memory. When MTL_RXP_Indirect_Acc_Control_Status[RXPEIEE] = 1, it indicates that on the basis of the value encoded in this field following errors are inserted: 00b - Insert 1 bit error 01b - Insert 2 bit errors 10b - Insert 3 bit errors 11b - Insert 1 bit error in address field
20 RXPEIEE	ECC Inject Error Enable for Rx Parser Memory Indicates whether the ECC inject error for receive parser memory is enabled. If this field is 1, it enables the ECC error injection feature. When this field becomes 0, it disables the ECC error injection feature.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
19-17 —	Reserved.
16 WRRDN	Read Write Control Controls the read and write operation to the receive parser memory. If this field is 1, it indicates the write operation to the receive parser memory. When this field is 0, it indicates the read operation to the receive parser memory. 0b - Read operation to the receive parser memory 1b - Write operation to the receive parser memory
15-8 —	Reserved.
7-0 ADDR	FRP Instruction Table Offset Address Indicates the ADDR of the 32-bit entry in receive parser instruction table. Each entry has 128-bit (4x32-bit words). The X depends on the configurable DWC_EQOS_FRP_ENTRIES. If DWC_EQOS_FRP_ENTRIES = 256 , then X = 9 If DWC_EQOS_FRP_ENTRIES = 128 , then X = 8 If DWC_EQOS_FRP_ENTRIES = 64, then X = 7

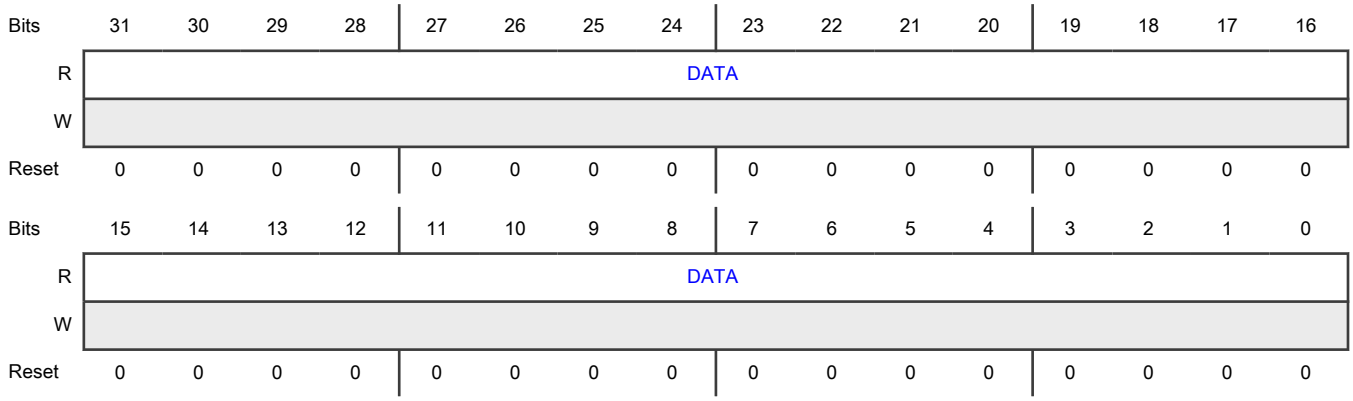
72.18.206 MTL Rx Parser Indirect Access Data (MTL_RXP_Indirect_Acc_Data)

Offset

Register	Offset
MTL_RXP_Indirect_Acc_Data	CB4h

Function
Provides the data associated to indirect access to receive parser memory.

Diagram



Fields

Field	Function
31-0 DATA	FRP Instruction Table Write/Read Data You must write this register before issuing any write command. After read command, when MTL_RXP_Indirect_Acc_Control_Status[STARTBUSY] = 0, the hardware provides the read data from the receive parser memory for read operation.

72.18.207 MTL ECC Control (MTL_ECC_Control)

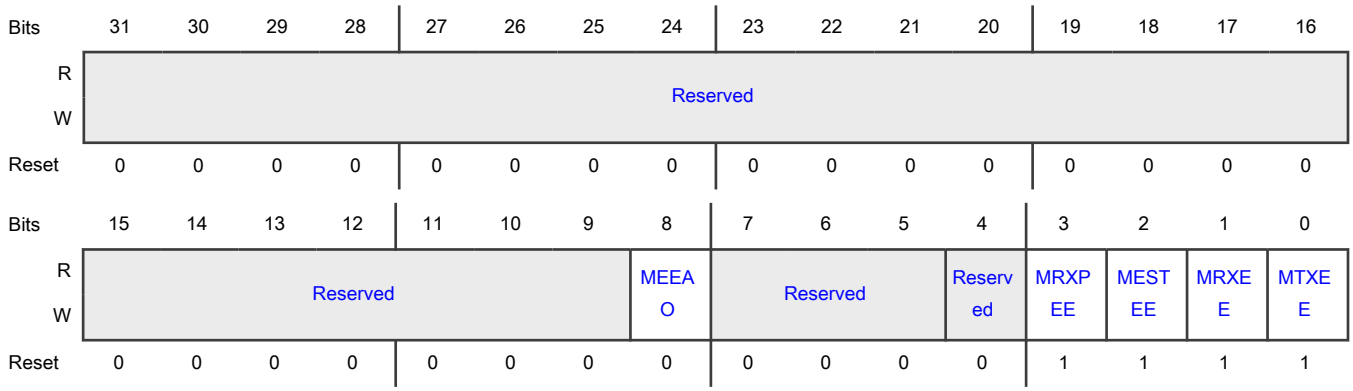
Offset

Register	Offset
MTL_ECC_Control	CC0h

Function

Establishes the operating mode of ECC related to MTL memories.

Diagram



Fields

Field	Function
31-9 —	Reserved.
8 MEEAO	MTL ECC Error Address Status Over-ride Enables or disables the MTL ECC error address status over-ride. When this field is 1, it indicates that MTL_ECC_Err_Addr_Status[EUEAS] and MTL_ECC_Err_Addr_Status[ECEAS] hold the last valid address where the error is detected. When this field becomes 0, it indicates that MTL_ECC_Err_Addr_Status[EUEAS] and MTL_ECC_Err_Addr_Status[ECEAS] hold the first address where the error is detected. 0b - Disable 1b - Enable
7-5 —	Reserved.
4 —	Reserved.
3 MRXPEE	MTL Rx Parser ECC Enable Indicates whether the MTL receive parser ECC is enabled. If this field is 1, it enables the ECC feature for receive parser memory. When this field is 0, it disables the ECC feature for receive parser memory. 0b - Disabled 1b - Enabled
2 MESTEE	MTL EST ECC Enable Indicates whether the MTL EST ECC is enabled. If this field is 1, it enables the ECC feature for EST memory. When this field is 0, it disables the ECC feature for EST memory. 0b - Disabled 1b - Enabled
1 MRXEE	MTL Rx FIFO ECC Enable Indicates whether the MTL Rx FIFO ECC is enabled. If this field is 1, it enables the ECC feature for MTL receive FIFO memory. When this field is 0, it disables the ECC feature for MTL receive FIFO memory. 0b - Disabled 1b - Enabled
0	MTL Tx FIFO ECC Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
MTXEE	Indicates whether the MTL Tx FIFO ECC is enabled. If this field is 1, it enables the ECC feature for MTL Tx FIFO memory. When field is 0, it disables the ECC feature for MTL Tx FIFO memory. 0b - Disabled 1b - Enabled

72.18.208 MTL Safety Interript Status (MTL_Safety_Interrupt_Status)

Offset

Register	Offset
MTL_Safety_Interrupt_Status	CC4h

Function

Provides safety interrupt status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														MEUIS	MECIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved Indicates whether the MAC safety uncorrectable interrupt status is detected. 0b - Not detected 1b - Detected
30-2	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 MEUIS	MTL ECC Uncorrectable Error Interrupt Status Indicates whether the MTL ECC uncorrectable error interrupt status is detected. Indicates that an uncorrectable error interrupt event in the MTL ECC safety feature. The application must read MTL_ECC_Interrupt_Status to get the exact cause of the interrupt. 0b - Not detected 1b - Detected
0 MECIS	MTL ECC Correctable Error Interrupt Status Indicates whether the MTL ECC correctable error interrupt status is detected. Indicates that a correctable error interrupt event in the MTL ECC safety feature. The application must read MTL_ECC_Interrupt_Status to get the exact cause of the interrupt . 0b - MTL ECC Correctable error Interrupt Status not detected 1b - MTL ECC Correctable error Interrupt Status detected

72.18.209 MTL ECC Interrupt Enable (MTL_ECC_Interrupt_Enable)

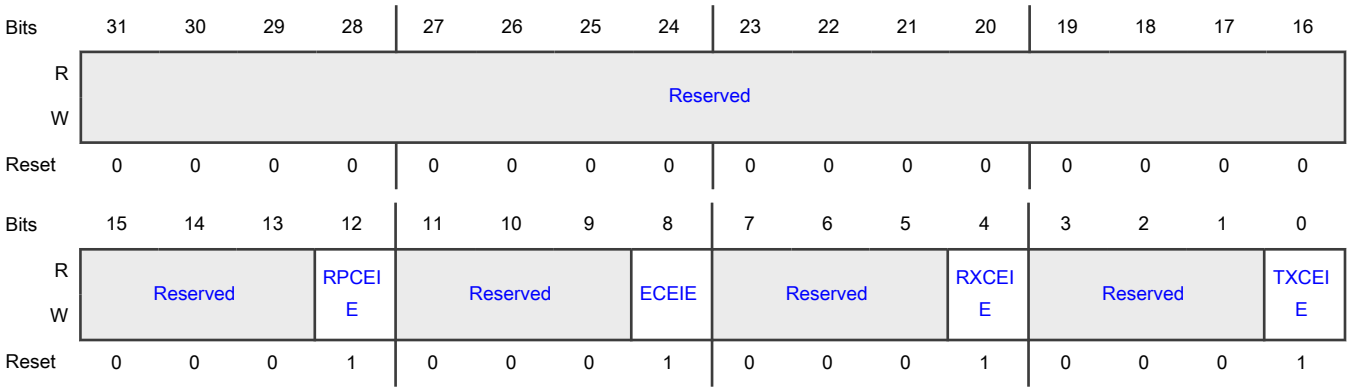
Offset

Register	Offset
MTL_ECC_Interrupt_Enable	CC8h

Function

Provides enable bits for the ECC interrupts.

Diagram



Fields

Field	Function
31-13 —	Reserved.
12 RPCEIE	Rx Parser Memory Correctable Error Interrupt Enable Indicates whether the Rx parser memory correctable error interrupt is enabled. When this field is 1, it generates an interrupt when an uncorrectable error is detected at the Rx parser memory interface. It is indicated in MTL_ECC_Interrupt_Status[RPCES]. When this field becomes 0, it indicates that this event does not generates an interrupt. 0b - Disabled 1b - Enabled
11-9 —	Reserved.
8 ECEIE	EST Memory Correctable Error Interrupt Enable Indicates whether the EST memory correctable error interrupt is enabled. When this field is 1, it generates an interrupt when a correctable error is detected at the MTL EST memory interface. It is indicated in MTL_ECC_Interrupt_Status[ECES]. When this field becomes 0, it indicates that this event does not generates an interrupt. 0b - Disabled 1b - Enabled
7-5 —	Reserved.
4 RXCEIE	Rx Memory Correctable Error Interrupt Enable Indicates whether the Rx memory correctable error interrupt is enabled. When this field is 1, it generates an interrupt when a correctable error is detected at the MTL Rx memory interface. It is indicated in MTL_ECC_Interrupt_Status[RXCES]. When this field becomes 0, it indicates that this event does not generates an interrupt. 0b - Disabled 1b - Enabled
3-1 —	Reserved.
0 TXCEIE	Tx Memory Correctable Error Interrupt Enable Indicates whether the Tx memory correctable error interrupt is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1, it generates an interrupt when a correctable error is detected at the MTL Tx memory interface. It is indicated in MTL_ECC_Interrupt_Status[TXCES]. When this field becomes 0, it indicates that this event does not generates an interrupt. 0b - Disabled 1b - Enabled

72.18.210 MTL ECC Interrupt Status (MTL_ECC_Interrupt_Status)

Offset

Register	Offset
MTL_ECC_Interrupt_Status	CCCh

Function
Provides MTL ECC Interrupt Status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	RPUES	RPAMS	RPCES	Reserved	EUES	EAMS	ECES	Reserved	RXUES	RXAMS	RXCES	Reserved	TXUES	TXAMS	TXCES
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved.
14 RPUES	Rx Parser Memory Uncorrectable Error Status Indicates whether the Rx parser memory uncorrectable error status is detected. When this field is 1, it indicates that an uncorrectable error is detected at Rx parser memory interface.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected
13 RPAMS	MTL Rx Parser Memory Address Mismatch Status Indicates whether the MTL Rx parser memory address mismatch status is detected. When this field is 1, it indicates that an address mismatch is found for Rx parser memory's address bus. 0b - Not detected 1b - Detected
12 RPCES	MTL Rx Parser Memory Correctable Error Status Indicates whther the MTL Rx parser memory correctable error status is detected. When this field is 1, it indicates that a correctable error is detected at RX parser memory interface. 0b - Not detected 1b - Detected
11 —	Reserved.
10 EUES	MTL EST Memory Uncorrectable Error Status Indicates whether the MTL EST memory uncorrectable error status is detected. When this field is 1, it indicates that an uncorrectable error is detected at MTL EST memory interface. 0b - Not detected 1b - Detected
9 EAMS	MTL EST Memory Address Mismatch Status Indicates whether the MTL EST memory address mismatch status is detected. When this field is 1, it indicates that an address mismatch is found for MTL EST memory's address bus. 0b - Not detected 1b - Detected
8 ECES	MTL EST Memory Correctable Error Status Indicates whether the MTL EST memory correctable error status is detected. When this field is 1, it indicates that a correctable error is detected at the MTL EST memory. 0b - Not detected 1b - Detected
7 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 RXUES	MTL Rx Memory Uncorrectable Error Status Indicates whether the MTL Rx memory uncorrectable error status is detected. When this field is 1, it indicates that an uncorrectable error is detected at the MTL Rx memory interface. 0b - Not detected 1b - Detected
5 RXAMS	MTL Rx Memory Address Mismatch Status Indicates whether the MTL Rx memory address mismatch status is detected. When this field is 1, it indicates that an address mismatch is found for MTL Rx memory's address bus. 0b - Not detected 1b - Detected
4 RXCES	MTL Rx memory Correctable Error Status Indicates whether the MTL Rx memory correctable error status is detected. When this field is 1, it indicates that a correctable error is detected at the MTL Rx memory. 0b - Not detected 1b - Detected
3 —	Reserved.
2 TXUES	MTL Tx Memory Uncorrectable Error Status Indicates whether the MTL Tx memory uncorrectable error status is detected. When this field is 1, it indicates that an uncorrectable error is detected at the MTL TX memory interface. 0b - Not detected 1b - Detected
1 TXAMS	MTL Tx Memory Address Mismatch Status Indicates whether the MTL Tx memory address mismatch status is detected. When this field is 1, it indicates that an address mismatch is found for MTL Tx memory's address bus. 0b - Not detected 1b - Detected
0 TXCES	MTL Tx Memory Correctable Error Status Indicates whether the MTL Tx memory correctable error status is detected. When this field is 1, it indicates that a correctable error is detected at the MTL Tx memory. 0b - Not detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Detected

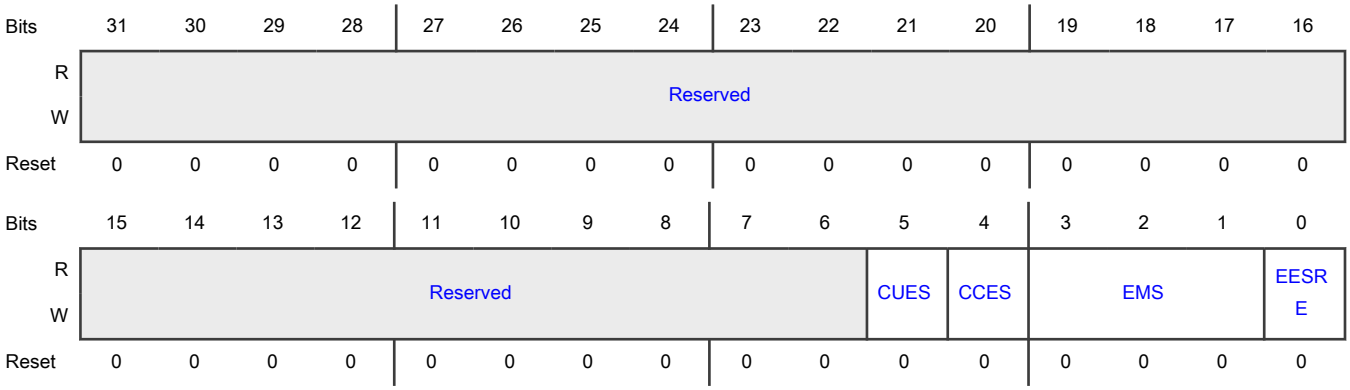
72.18.211 MTL ECC Error Status (MTL_ECC_Err_Sts_Rctl)

Offset

Register	Offset
MTL_ECC_Err_Sts_Rctl	CD0h

Function
Establishes the control for ECC error status capture.

Diagram



Fields

Field	Function
31-6 —	Reserved.
5 CUES	Clear Uncorrectable Error Status Indicates whether the clear uncorrectable error status is detected. When this field and MTL_ECC_Err_Sts_Rctl[EESRE] is 1, it indicates that based on the MTL_ECC_Err_Sts_Rctl[EMS], the respective memory's uncorrectable error address and uncorrectable error count values are cleared upon reading. When all the error status values are cleared hardware resets this field. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 CCES	Clear Correctable Error Status Indicates whether the clear correctable error status is detected. When this field and MTL_ECC_Err_Sts_Rctl[EESRE] is 1, it indicates that based on the MTL_ECC_Err_Sts_Rctl[EMS],, the respective memory's correctable error address and correctable error count values are cleared upon reading. When all the error status values are cleared, hardware resets this field. 0b - Not detected 1b - Detected
3-1 EMS	MTL ECC Memory Selection Provides the memory selection encoding. When MTL_ECC_Err_Sts_Rctl[EESRE] is 1, this field indicates which memory's error status value to be read. The memory selection encoding is as described below. 000b - MTL Tx memory 001b - MTL Rx memory 010b - MTL EST memory 011b - MTL Rx Parser memory 100b - DMA TSO memory
0 EESRE	MTL ECC Error Status Read Enable Indicates whether the MTL ECC error status read is enabled. When this field is 1, it indicates that based on MTL_ECC_Err_Sts_Rctl[EMS], the respective memory's error status values are captured as described below: • The correctable and uncorrectable error count values are captured into MTL_ECC_Err_Cnt_Status. • The address location of correctable and uncorrectable errors are captured into MTL_ECC_Err_Addr_Status. When all the status values are captured into MTL_ECC_Err_Cnt_Status and MTL_ECC_Err_Addr_Status, hardware resets this field. 0b - Disabled 1b - Enabled

72.18.212 MTL ECC Error Adress Status (MTL_ECC_Err_Addr_Status)

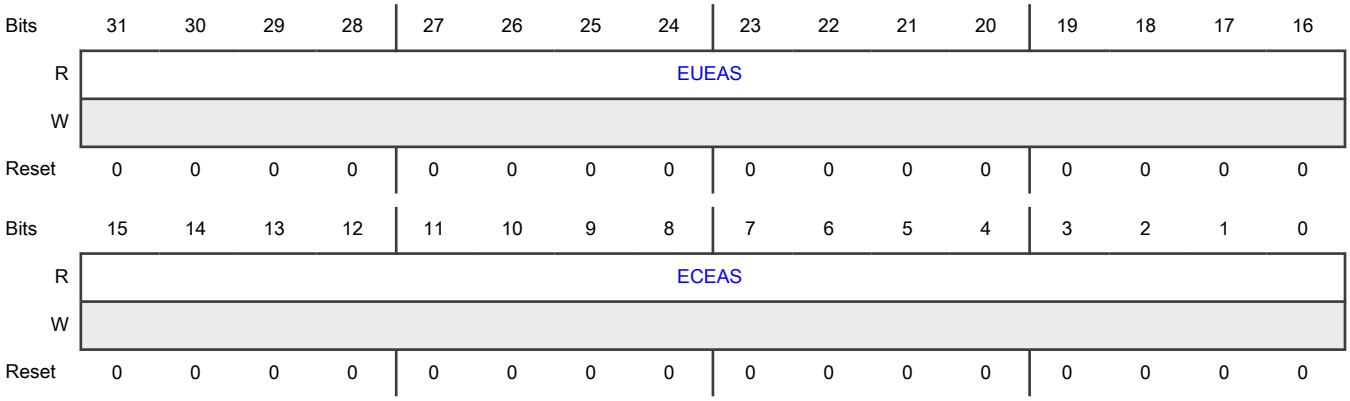
Offset

Register	Offset
MTL_ECC_Err_Addr_Status	CD4h

Function

Provides the memory addresses for the correctable and uncorrectable errors.

Diagram



Fields

Field	Function
31-16 EUEAS	MTL ECC Uncorrectable Error Address Status Based on the MTL_ECC_Err_Sts_Rctl[EMS], this field holds the respective memory's address locations for which an uncorrectable error or address mismatch is detected. When MTL_ECC_Control[MEEAO] = 1, this field holds the last valid address of memory for which either an uncorrectable error or an address mismatch is detected. When MTL_ECC_Control[MEEAO] = 0, this field holds the first address of the memory for which either an uncorrectable error or address mismatch is detected.
15-0 ECEAS	MTL ECC Correctable Error Address Status Based on the MTL_ECC_Err_Sts_Rctl[EMS], this field holds the respective memory's address locations for which a correctable error is detected. When MTL_ECC_Control[MEEAO] = 1, this field holds the last valid address of memory for which the correctable error or address mismatch is detected. When MTL_ECC_Control[MEEAO] = 0, this field holds the first address of the memory for which correctable error is detected.

72.18.213 MTL ECC Error Control Status (MTL_ECC_Err_Cntr_Status)

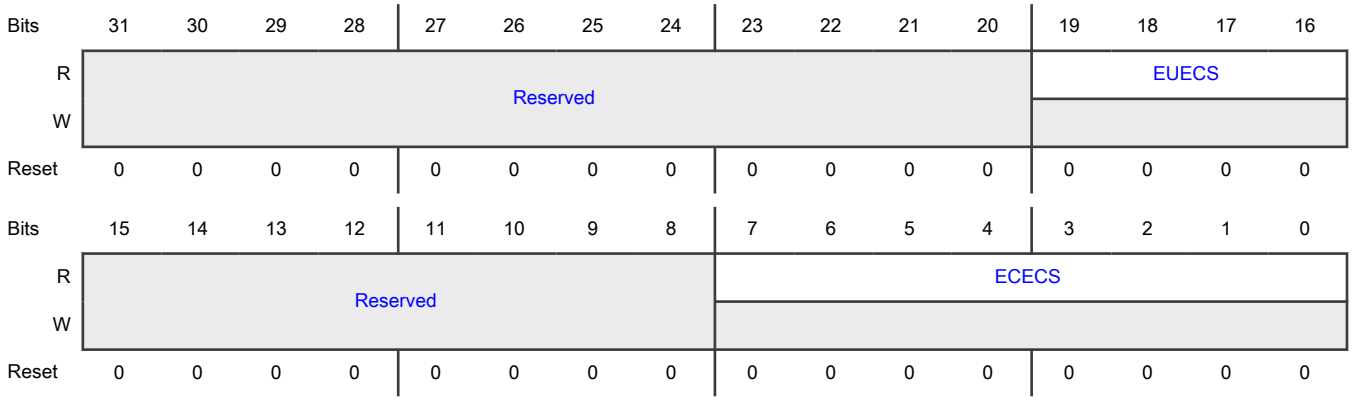
Offset

Register	Offset
MTL_ECC_Err_Cntr_Status	CD8h

Function

Provides ECC Error count for correctable and uncorrectable errors.

Diagram



Fields

Field	Function
31-20 —	Reserved.
19-16 EUECS	MTL ECC Uncorrectable Error Counter Status Based on the EMS field of MTL_ECC_Err_Cntr_Rctl register, this field holds the respective memory's uncorrectable error count value.
15-8 —	Reserved.
7-0 ECECS	MTL ECC Correctable Error Counter Status Based on the EMS field of MTL_ECC_Err_Cntr_Rctl register, this field holds the respective memory's correctable error count value.

72.18.214 MTL DPP Control (MTL_DPP_Control)

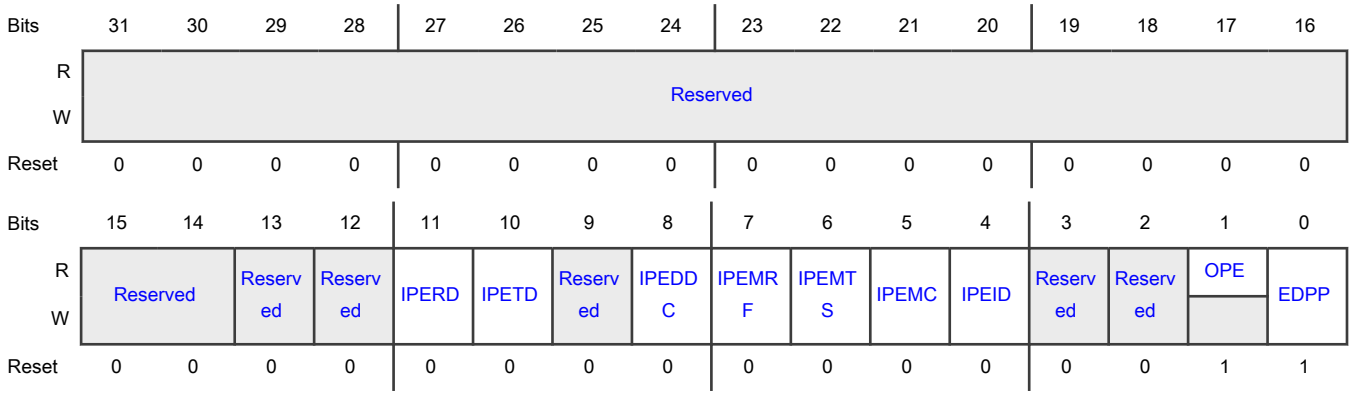
Offset

Register	Offset
MTL_DPP_Control	CE0h

Function

Establishes the operating mode of data parity protection and error injection.

Diagram



Fields

Field	Function
31-14 —	Reserved.
13 —	Reserved.
12 —	Reserved.
11 IPERD	Insert Parity error in Rx write-back Descriptor parity generator Indicates whether the insert parity error in receive write-back descriptor parity generator is enabled. When this field is 1, it flips the parity bit of first valid data to which the DMA Rx write-back descriptor parity generator(or at PG8 as shown in Receive data path parity protection diagram) generates. 0b - Disabled 1b - Enabled
10 IPETD	Insert Parity error in Tx write-back Descriptor parity generator Indicates whether the insert parity error in transit write-back descriptor parity generator is enabled. When this field is 1, it flips the parity bit of first valid data to which the DMA Tx write-back descriptor parity generator(or at PG4 as shown in Transmit data path parity protection diagram) generates. Hardware clears this field, when the respective parity bit flips. 0b - Disabled 1b - Enabled
9 —	Reserved.
8	Insert Parity Error in DMA DTX Control Word Parity Generator

Table continues on the next page...

Table continued from the previous page...

Field	Function
IPEDDC	Indicates whether the insert parity error in DMA DTX control word parity generator is enabled. When this field is 1, it flips the parity bit of first valid data to which the DMA DTX Control word parity generator (or at PG2 as shown in Transmit data path parity protection diagram) generates. Hardware clears this field, when the respective parity bit flips. 0b - Disabled 1b - Enabled
7 IPEMRF	Insert Parity Error in MTL Rx FIFO Read Control Parity Generator Inserts parity Eerror in MTL receive FIFO read control parity generator. When this field is 1, it flips the parity bit of first valid data to which the MTL Rx FIFO read control parity generator (or at PG7 as shown in Receive data path parity protection diagram) generates. Hardware clears this field, when the respective parity bit flips. 0b - Disabled 1b - Enabled
6 IPEMTS	Insert Parity Error in MTL Tx Status Parity Generator Indicates whether the insert parity error in MTL transit status parity generator is enabled. When this field is 1, it flips the parity bit of first valid data to which the MTL Tx Status parity generator (or at PG6 as shown in Transmit data path parity protection diagram) generates. Hardware clears this field, when the respective parity bit flips. 0b - Disabled 1b - Enabled
5 IPEMC	Insert Parity Error in MTL Checksum Parity Generator Indicates whether the insert parity error in MTL checksum parity generator is enabled. When this field is 1, it flips the parity bit of first valid data to which the MTL checksum parity generator (or at PG5 as shown in Transmit data path parity protection diagram) generates. Hardware clears this field, when the respective parity bit flips. 0b - Disabled 1b - Enabled
4 IPEID	Insert Parity Error in Interface Data Parity Generator Indicates whether the insert parity error in interface data parity generator is enabled. When this field is 1, it flips the parity bit of first valid input data to which the interface data parity generator (or at PG1 as shown in Transmit data path parity protection diagram) generates. Following are the input data bus on which parity bits are generated on the basis of configuration selected. In AHB Config, hrdata_i In AXI config, rdata_m_i

Table continues on the next page...

Table continued from the previous page...

Field	Function
	In DMA Config, mdc_rdata_i In MTL Config, ati_data_i Hardware clears this field, when the respective parity bit flips. 0b - Disabled 1b - Enabled
3 —	Reserved.
2 —	Reserved.
1 OPE	Odd Parity Enable Indicates whether an odd parity protection is enabled. When this field is 1, it enables an odd parity protection on all the external interfaces and when this field is 0, it enables an even parity protection on all the external interfaces. 0b - Disabled 1b - Enabled
0 EDPP	Enable Data path Parity Protection Enables or disables the data path parity protection. When this field is 1, it indicates that by generating and checking the parity on EQOS datapath it enables the parity protection for EQOS datapath . When this field is 0, it disables the parity protection for EQOS datapath. 0b - Disable 1b - Enable

72.18.215 MTL Tx Queue 0 Operation Mode (MTL_TxQ0_Operation_Mode)

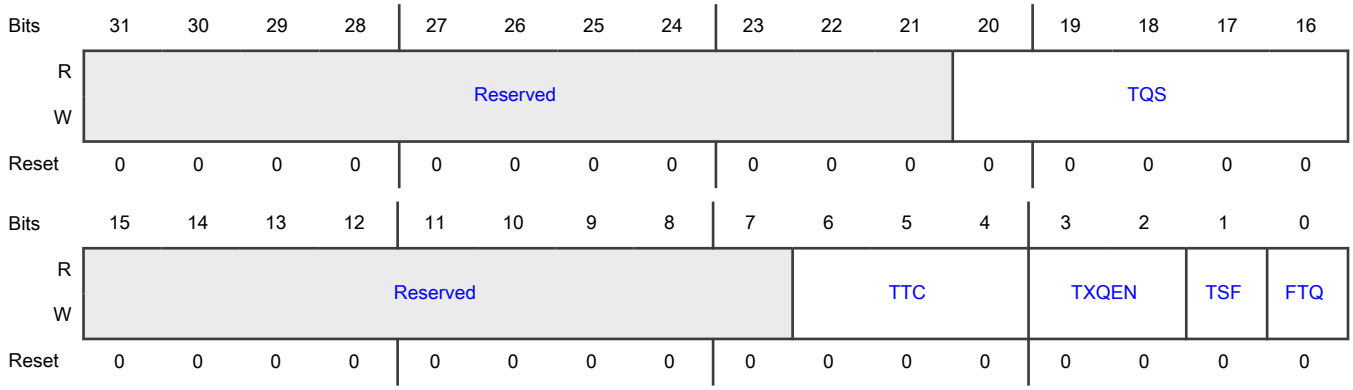
Offset

Register	Offset
MTL_TxQ0_Operation_Mode	D00h

Function

Establishes the transmit queue operating modes and commands.

Diagram



Fields

Field	Function
31-21 —	Reserved.
20-16 TQS	Transmit Queue Size Indicates the size of the allocated transmit queues in blocks of 256 bytes. This field is read-write only if the number of transit queues are more than one, the reset value is 0x0 and indicates 256 bytes size. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. You must program TQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes . In general, the size of the Queue = (TQS+1)*256 bytes. When the number of transit queue is one, the field is read-only and the reset value reflects the configured TX FIFO size in blocks of 256 bytes. The field's width depends on the transit memory size selected in your configuration. For example, if the memory size is 2048, the field's width is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
15-7 —	Reserved.
6-4 TTC	Transmit Threshold Control Controls the threshold level of the MTL transit queue. The transmission starts when the packet size within the MTL transit queue is larger than the threshold, also transmits full packets with length less than the threshold. These fields are used only when MTL_TxQ0_Operation_Mode[TSF] resets. 000b - 32 001b - 64 010b - 96 011b - 128 100b - 192 101b - 256

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - 384 111b - 512
3-2 TXQEN	Transmit Queue Enable Enables or disables the transmit queue 0. 2'b00 - Not enabled 2'b01 - Reserved 2'b10 - Enabled 2'b11 - Reserved This field is read only in single queue configurations and read write in multiple queue configurations. **NOTE** In multiple transit queue configuration, all the queues disables by default and programming this field enables the transit queue. 00b - Not enabled 01b - Enable in AV mode (Reserved in non-AV) 10b - Enabled 11b - Reserved
1 TSF	Transmit Store and Forward Indicates whether the transmit store and forward is enabled. When this field is 1, it indicates that the transmission starts when a full packet resides in the MTL transit queue and the TTC values specified in Bits[6:4] of this register are ignored. This field must be changed only when you stop the transmission. 0b - Disabled 1b - Enabled
0 FTQ	Flush Transmit Queue Indicates whether the flush transmit queue is enabled. When this field is 1, it indicates that the transit queue controller logic resets to its default values. Therefore, all the data in the transit queue is lost or flushed. After the flushing operation completes this field resets internally. You must not write to the MTL_TxQ1_Operation_Mode until this field resets. The data to which the MAC transmitter has already accepted is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. **NOTE** The flush operation completes only when the transit queue is empty and the application accepts the pending transit status of all the transmitted packets. To complete the flush operation, the PHY Tx clock (clk_tx_i) must be active.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It clears automatically. Writing 1 sets this field and writing 0 has no effect. 0b - Disabled 1b - Enabled

72.18.216 MTL Tx Queue 0 Underflow (MTL_TxQ0_Underflow)

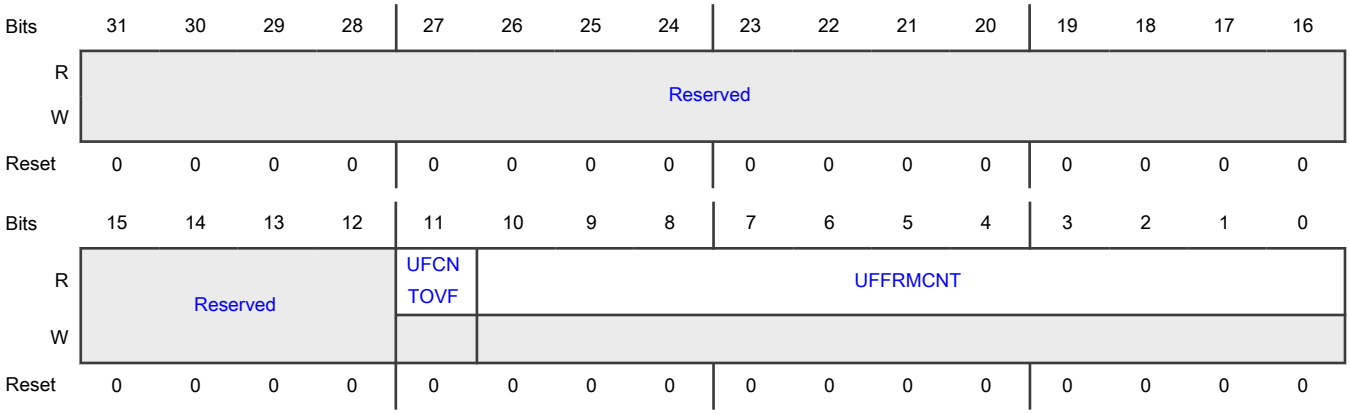
Offset

Register	Offset
MTL_TxQ0_Underflow	D04h

Function

Contains the counter for packets aborted because of transmit queue underflow and packets missed because of receive queue packet flush.

Diagram



Fields

Field	Function
31-12 —	Reserved.
11 UFCNTOVF	Overflow Bit for Underflow Packet Counter Indicates whether the overflow is detected for underflow packet counter.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is 1, when the transit queue underflow packet counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter resets to all-zeros and this field indicates that the rollover occured. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
10-0 UFFRMCNT	Underflow Packet Counter Indicates the number of packets aborted by the controller because of transit queue underflow. This counter increments each time the MAC aborts outgoing packet because of underflow. The counter clears when this register is read with mci_be_i[0] at 1'b1. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

72.18.217 MTL Tx Queue 0 Debug (MTL_TxQ0_Debug)

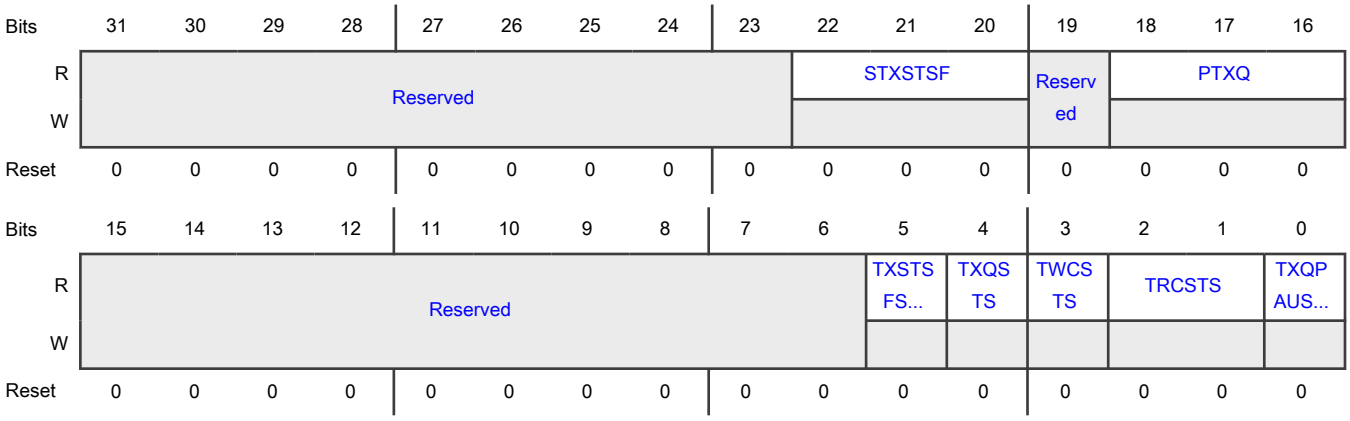
Offset

Register	Offset
MTL_TxQ0_Debug	D08h

Function

Provides the debug status of various blocks related to the transmit queue.

Diagram



Fields

Field	Function
31-23 —	Reserved.
22-20 STXSTSF	Number of Status Words in Tx Status FIFO of Queue Indicates the current number of status in the transit status FIFO of this queue. This field does not reflect the number of status words in transit status FIFO, when MTL_Operation_Mode[DTXSTS] = 1.
19 —	Reserved.
18-16 PTXQ	Number of Packets in the Transmit Queue Indicates the current number of packets in the transit queue. This field does not reflect the number of packets in the transmit queue, when MTL_Operation_Mode[DTXSTS] = 1.
15-6 —	Reserved.
5 TXSTSFSTS	MTL Tx Status FIFO Full Status Indicates whether the MTL transit status and FIFO full status is detected. When this field is 1, it indicates that the MTL transit status FIFO is full. Therefore, MTL cannot accept any more packets for transmission. 0b - Not detected 1b - Detected
4 TXQSTS	MTL Tx Queue Not Empty Status Indicates whether the MTL transit queue not empty status is detected. When this field is high, it indicates that the MTL transit queue is not empty and some data is left for transmission. 0b - Not detected 1b - Detected
3 TWCSTS	MTL Tx Queue Write Controller Status Indicates whether the MTL transit queue write controller status is detected. When high, this field indicates that the MTL transit queue write controller is active, and transfers the data to the transit queue. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
2-1 TRCSTS	MTL Tx Queue Read Controller Status Indicates the state of the transit queue read controller. 00b - Idle state 01b - Read state (transferring data to the MAC transmitter) 10b - Waiting for pending transit status from the MAC transmitter 11b - Flushing the transit queue because of the packet abort request from the MAC
0 TXQPAUSED	Transmit Queue in Pause Indicates whether the transmit queue in pause is detected. When this field is 1 and the receive flow control is enabled, it indicates that the transit queue is in the pause condition (in the full-duplex only mode) because of the following scenario: • Receives PFC packet with the priorities assigned to the transit queue when PFC is enabled. • Receives 802.3x pause packet when PFC is disabled. 0b - Not detected 1b - Detected

72.18.218 MTL Tx Queue 0 ETS Status (MTL_TxQ0_ETS_Status)

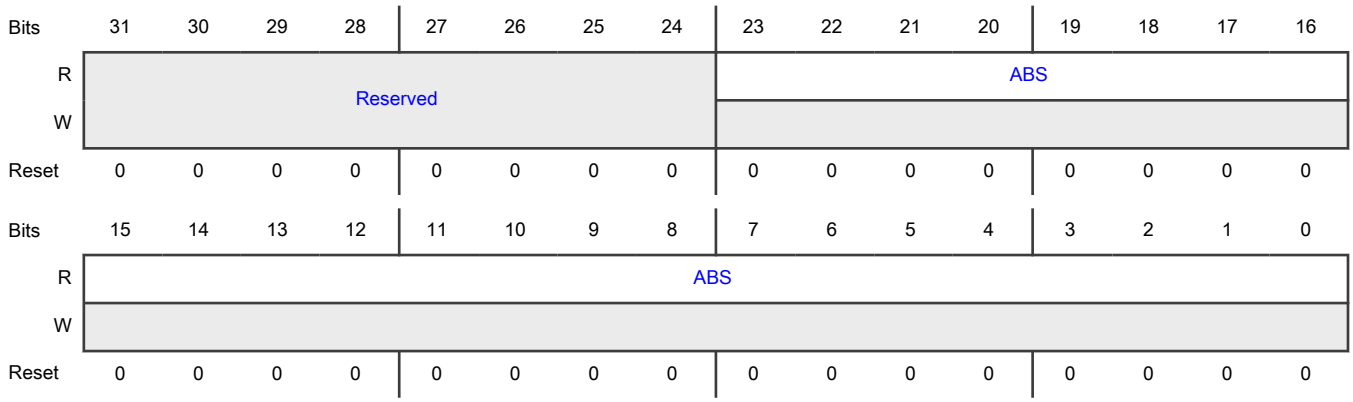
Offset

Register	Offset
MTL_TxQ0_ETS_Status	D14h

Function

Provides the average traffic transmitted in queue 0.

Diagram



Fields

Field	Function
31-24 —	Reserved.
23-0 ABS	Average Bits per Slot Contains the average transmitted bits per slot. Computes over every 10 million bit times slot (4 ms in 2500 Mbit/s; 10 ms in 1000 Mbit/s; 100 ms in 100 Mbit/s), when the DCB operation enables for queue 0. The maximum value is 0x989680.

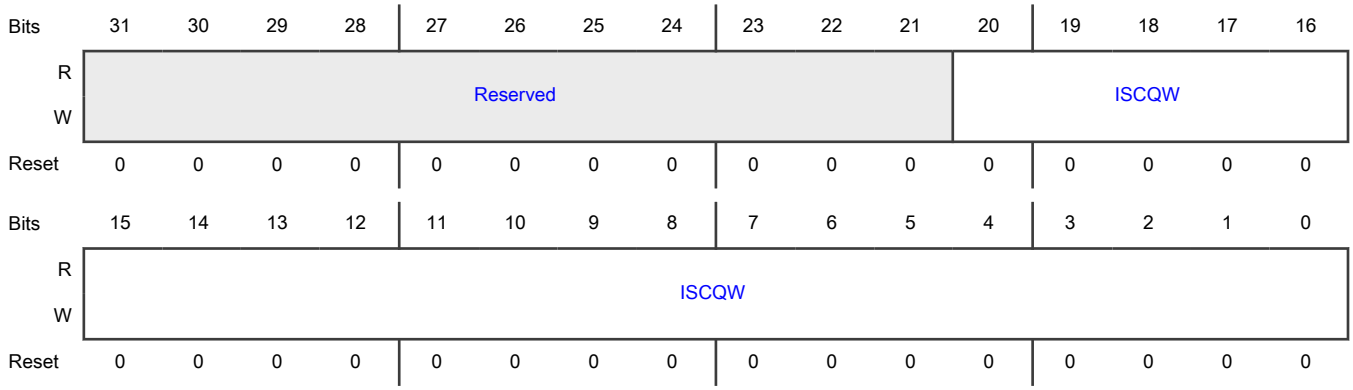
72.18.219 MTL Tx Queue Quantum Weight (MTL_TxQ0_Quantum_Weight)

Offset

Register	Offset
MTL_TxQ0_Quantum_Weight	D18h

Function
Contains the quantum value for Deficit Weighted Round Robin (DWRR), weights for the Weighted Round Robin (WRR), and Weighted Fair Queuing (WFQ) for queue 0.

Diagram



Fields

Field	Function
31-21 —	Reserved.
20-0	Quantum or Weights

Table continues on the next page...

Table continued from the previous page...

Field	Function
ISCQW	Contains the quantum value in bytes which is added to credit during every queue scanning cycle, when DCB operation enables with DWRR algorithm for queue 0 traffic. The maximum value is 0x1312D0 bytes. Contains the weight for this queue, when DCB operation enables with WFQ algorithm for queue 0 traffic. The maximum value is 0x3FFF where weight of 0 indicates 100% bandwidth. Write 0 to bits[20:14]. The higher the programmed weights, the lesser is the bandwidth that is allocated for the particular transmit queue. This is because the weights compute the packet finish time (weights*packet_size). The lesser the finish time, the higher is the probability of the packet getting scheduled first and consuming more bandwidth. Contains the weight for this queue, when DCB operation or generic queuing operation enables with WRR algorithm for queue 0 traffic. The maximum value is 0x64. Write 0 to bits [20:7].

72.18.220 MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status)

Offset

Register	Offset
MTL_Q0_Interrupt_Control_Status	D2Ch

Function

Contains the interrupt enable and status bits for the queue 0 interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved							RXOIE	Reserved							RXOVFIS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						ABPSIE	TXUIE	Reserved						ABPSIS	TXUNFIS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 RXOIE	Receive Queue Overflow Interrupt Enable Enables or disables the receive queue overflow interrupt. When this field is 1, it enables the receive queue overflow interrupt. When this field becomes 0, it disables the receive queue overflow interrupt. 0b - Disabled 1b - Enabled
23-17 —	Reserved.
16 RXOVFIS	Receive Queue Overflow Interrupt Status Indicates whether the status of receive queue overflow interrupt is detected. This bit indicates that the receive queue had an overflow when you receive the packet. If you transfer a partial packet to the application, the overflow status sets in RDES3[21]. This fiels is 0 when the application writes 1 to it. Access restriction apply to this field. It becomes 0 automatically on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
15-10 —	Reserved.
9 ABPSIE	Average Bits Per Slot Interrupt Enable Enables or disables the average bits per slot interrupt. When this field is 1, it indicates that when you update the average bits per slot status, the MAC asserts the sbd_intr_o or mci_intr_o interrupt. When this field is 0, it indicates that the interrupt do not assert for such an event. 0b - Disabled 1b - Enabled
8 TXUIE	Transmit Queue Underflow Interrupt Enable Enables or disables transmit queue underflow interrupt. When this field is 1, it enables the transmit queue underflow interrupt. When this field becomes 0, it disables the transmit queue underflow interrupt. 0b - Disabled 1b - Enabled
7-2	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 ABPSIS	Average Bits Per Slot Interrupt Status Indicates whether the status of average bits per slot interrupt is detected. When this field is 1, it indicates that the MAC has updated the ABS value. This field is 0 when the application writes 1 it. Access restriction apply to this field. It becomes 0 automatically on an internal event occurrence. Writing 1 sets this field and writing 0 has no effect. 0b - Not detected 1b - Detected
0 TXUNFIS	Transmit Queue Underflow Interrupt Status Indicates whether the status of transmit queue underflow interrupt is detected. This field indicates that, when you transmit the packet,the transmit queue had an underflow. Suspend the transmission and write 1 to an underflow error TDES3[2]. This field is 0, when the application writes 1 to it. Access restriction apply to this field. It becomes 0 automatically on an internal event occurrence. Writing 1 sets this field and writing 0 has no effect. 0b - Not detected 1b - Detected

72.18.221 MTL Rx Queue 0 Operation Mode (MTL_RxQ0_Operation_Mode)

Offset

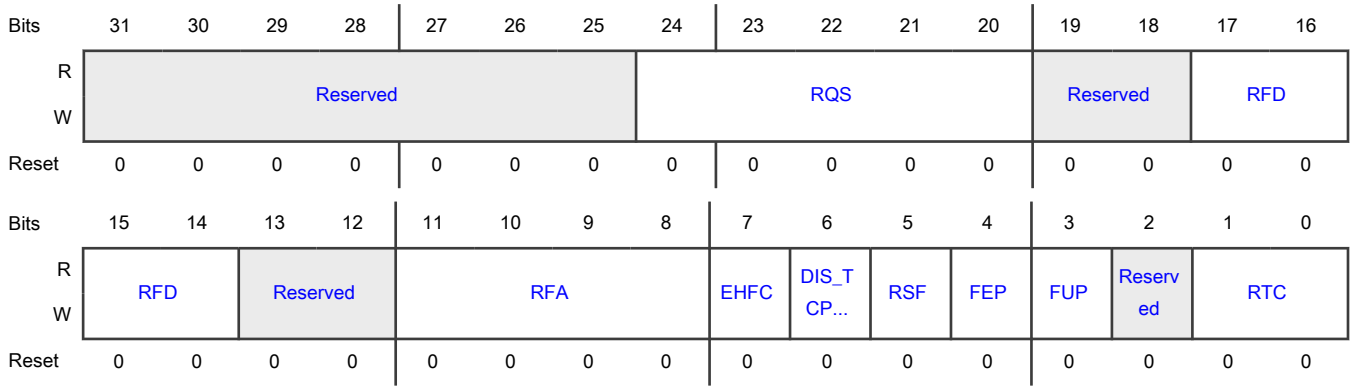
Register	Offset
MTL_RxQ0_Operation_Mode	D30h

Function

Establishes the receive queue operating modes and command.

The RFA and RFD fields are not backward compatible with the RFA and RFD fields of 4.00a release

Diagram



Fields

Field	Function
31-25 —	Reserved.
24-20 RQS	Receive Queue Size Indicates the size of the allocated receive queues in blocks of 256 bytes. The MTL_RxQ0_Operation_Mode[RQS] is read-write only if the number of receive queues are more than one, the reset value is 0x0 and indicates size of 256 bytes. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. You must program RQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes. In general, the size of the Queue = (RQS+1)*256 bytes. When the number of receive queues is one, the field is read-only and the configured receive FIFO size in blocks of 256 bytes is reflected in the reset value. The field width depends on the receive memory size selected in your configuration. For example, if the memory size is 2048, the field width is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
19-18 —	Reserved.
17-14 RFD	Threshold for Deactivating Flow Control (in half-duplex and full-duplex modes) Controls the threshold (fill-level of Rx queue) at which the flow control is de-asserts after activation. 0 - Full minus 1 KB, that is, FULL 1 KB 1 - Full minus 1.5 KB, that is, FULL 1.5 KB 2 - Full minus 2 KB, that is, FULL 2 KB 3 - Full minus 2.5 KB, that is, FULL 2.5 KB ... 14 - Full minus 8 KB, that is, FULL 8 KB 15 - Full minus 8.5 KB, that is, FULL 8.5 KB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	De-assertion is effective only after flow control is asserted. **NOTE** You must program the value in such a way that the threshold is a positive number. When MTL_RxQ0_Operation_Mode[EHFC] = 1, these values are applicable only when the receive queue size which the MTL_RxQ0_Operation_Mode[RQS] determines is equal to or greater than 4 KB. For a given queue size, the values ranges between 0 and the encoding for FULL minus (QSIZE - 0.5 KB) and all other values are illegal. Here the term FULL and QSIZE refers to the queue size which the MTL_RxQ0_Operation_Mode[RQS] determines. The field width depends on RX FIFO size selected during the configuration. Remaining bits are reserved and read only.
13-12 —	Reserved.
11-8 RFA	Threshold for Activating Flow Control (in half-duplex and full-duplex) Controls the threshold (fill-level of receive queue) at which the flow control is activated. See MTL_RxQ0_Operation_Mode[RFD] for more information on encoding for this field.
7 EHFC	Enable Hardware Flow Control Enables or disables hardware flow control. If this field is 1, it enables the flow control signal operation, on the basis of the fill-level of receive queue. When this field becomes 0, it disables the flow control operation. 0b - Disable 1b - Enable
6 DIS_TCP_EF	Disable Dropping of TCP/IP Checksum Error Packets Enables or disable dropping of TCP or IP checksum error packets. When this field is 1, it indicates that the MAC does not drop the packets which only have the errors which the receive checksum offload engine detects. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. When this field becomes 0, it indicates that all error packets are dropped if MTL_RxQ0_Operation_Mode[FEP] resets. 0b - Enable 1b - Disable
5 RSF	Receive Queue Store and Forward Indicates whether the receive queue store and forward is enabled. When this field is 1, it indicates that the module reads a packet from the receive queue only after the complete packet has been written to it, ignores the MTL_RxQ0_Operation_Mode[RTC].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field becomes 0, it indicates that the receive queue operates in the Threshold (cut-through) mode, subject to the threshold which the MTL_RxQ0_Operation_Mode[RTC] specifies. 0b - Disabled 1b - Enabled
4 FEP	Forward Error Packets Indicates whether the forward error packets are enabled. When this field becomes 0, it indicates that the receive queue drop packets with an error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, the packet does not drop if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode). When this field is 1, it indicates that all packets except the runt error packets are forwarded to the application or DMA. The packet is dropped irrespective of the setting of this field, if MTL_RxQ0_Operation_Mode[RSF] is 1 and the receive queue overflows when a partial packet is written. However, if MTL_RxQ0_Operation_Mode[RSF] becomes 0 and the receive queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA. 0b - Disabled 1b - Enabled
3 FUP	Forward Undersized Good Packets Indicates whether the forward undersized good packets are enabled. When this field is 1, it indicates that the receive queue forwards the undersized good packets (packets with no error and length less than 64 bytes), including pad-bytes and CRC. When this field becomes 0, it indicates that the receive queue drops all packets of less than 64 bytes, unless a packet is already transferred because of the lower value of receive threshold, for example, MTL_RxQ0_Operation_Mode[RTC] = 01. 0b - Disabled 1b - Enabled
2 —	Reserved.
1-0 RTC	Receive Queue Threshold Control Controls the threshold level of the MTL receive queue (in bytes). The received packet is transferred to the application or DMA when the packet size within the MTL receive queue is larger than the threshold. In addition, automatically tranfer full packets with length less than the threshold. This field is valid only when MTL_RxQ0_Operation_Mode[RSF] = 0. This field is ignored when MTL_RxQ0_Operation_Mode[RSF] = 1. 00b - 64

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - 32 10b - 96 11b - 128

72.18.222 MTL Rx Queue Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt)

Offset

Register	Offset
MTL_RxQ0_Missed_Packet_Overflow_Cnt	D34h

Function

Contains the counter for packets missed because of receive queue packet flush and packets discarded because of receive queue overflow.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				MISCNTO...	MISPKTCNT										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				OVFCNTO...	OVFPKTCNT										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved.
27 MISCNTOVF	Missed Packet Counter Overflow Bit Indicates whether the missed packet counter overflow is detected. When this field is 1, it indicates that the receive queue missed packet counter has crossed the maximum limit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
26-16 MISPKTCNT	Missed Packet Counter Indicates the number of packets this module has missed because the application asserts ari_pkt_flush_i[] for this queue. This counter resets when this register reads with mci_be_i[0] at 1b1. This counter increments by 1 when the DMA discards the packet because of buffer unavailability. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.
15-12 —	Reserved.
11 OVFCNTOVF	Overflow Counter Overflow Bit Indicates whether the counter overflow is detected. When this field is 1, it indicates that the receive queue overflow packet counter field has crossed the maximum limit. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
10-0 OVFPKTCNT	Overflow Packet Counter Indicates the number of packets which this module discards because of receive queue overflow. This counter increments each time the module discards an incoming packet because of overflow. This counter resets when this register reads with mci_be_i[0] at 1'b1. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

72.18.223 MTL Rx Queue 0 Debug (MTL_RxQ0_Debug)

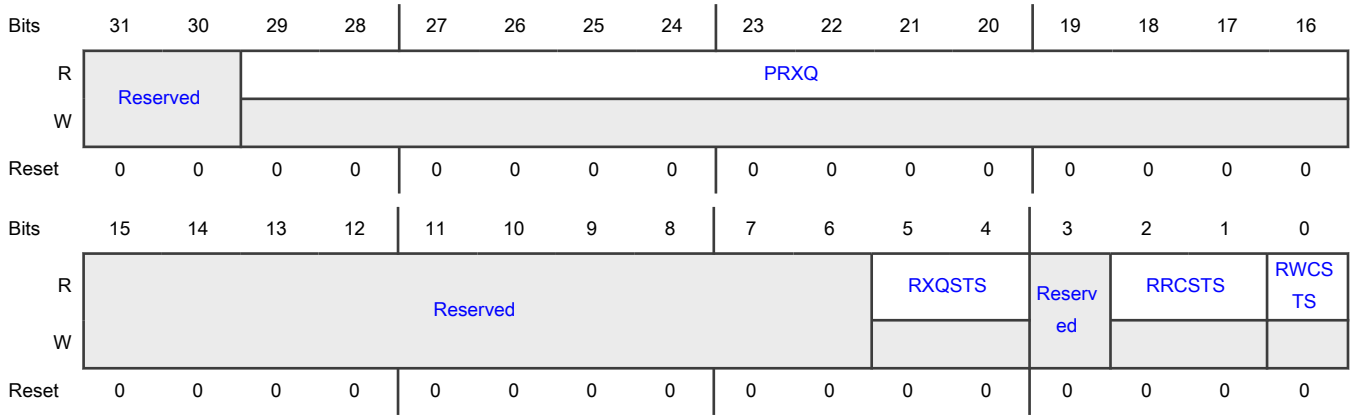
Offset

Register	Offset
MTL_RxQ0_Debug	D38h

Function

Provides the debug status of various blocks related to the receive queue.

Diagram



Fields

Field	Function
31-30 —	Reserved.
29-16 PRXQ	Number of Packets in Receive Queue Indicates the current number of packets in the receive queue. The theoretical maximum value for this field is 256KB/16B = 16K packets, that is, Max_Queue_Size/Min_Packet_Size.
15-6 —	Reserved.
5-4 RXQSTS	MTL Rx Queue Fill-Level Status Provides the status of the receive queue fill-level. 00b - Rx Queue empty 01b - Rx Queue fill-level below flow-control deactivate threshold 10b - Rx Queue fill-level above flow-control activate threshold 11b - Rx Queue full
3 —	Reserved.
2-1 RRCSTS	MTL Rx Queue Read Controller State Provides the state of the receive queue read controller. 00b - Idle state 01b - Reading packet data 10b - Reading packet status (or timestamp) 11b - Flushing the packet data and status

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 RWCSTS	MTL Rx Queue Write Controller Active Status Indicates whether the MTL receive queue write controller active status is detected. When high, this bit indicates that the MTL receive queue write controller is active, and it is transferring a received packet to the receive queue. 0b - Not detected 1b - Detected

72.18.224 MTL Rx Queue 0 Control 0 (MTL_RxQ0_Control)

Offset

Register	Offset
MTL_RxQ0_Control	D3Ch

Function

Controls the receive arbitration and passing of received packets to the application.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												RXQ_FRM...	RXQ_WEGT		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved.
3 RXQ_FRM_ARBIT	Receive Queue Packet Arbitration Indicates whether the receive queue packet arbitration is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1, it indicates that the module drives the packet data to the ARI interface such that the entire packet data of currently-selected queue transmits before switching to other queue. When this field becomes 0, it indicates that the module drives the packet data to the ARI interface such that the following amount of data of currently-selected queue transmits before switching to other queue: • PBL amount of data (indicated by ari_qN_pbl_i[]), or • Complete data of a packet The status and the timestamp are not a part of the PBL data. Therefore, the module drives the complete status (including timestamp status) during first PBL request for the packet (in store-and-forward mode) or the last PBL request for the packet (in Threshold mode). 0b - Disabled 1b - Enabled
2-0 RXQ_WEGT	Receive Queue Weight Indicates the weight assigned to receive queue 0. You must write a value to this field that is 1 less than the required queue weight. Therefore, if this field is 0, it indicates that the queue weight is 1, if this field is 1, it indicates that the queue weight is 2, and so on. You must use this weight to calculate the number of continuous PBL or packet requests, depending on the value of RXQ_FRM_ARBIT, allocated to the queue in an arbitration cycle. The field's value changes when the current service round completes or when there is a change from RAA=SP to RAA=WSP algorithm. This approach is required for a smooth transition. You must configure MTL Rx Queue 0 Control 0 (MTL_RxQ0_Control) before MTL Operation Mode (MTL_Operation_Mode) if you want to change the field's value before either of these two processes completes.

72.18.225 MTL Tx Queue 1 Operation Mode (MTL_TxQ1_Operation_Mode)

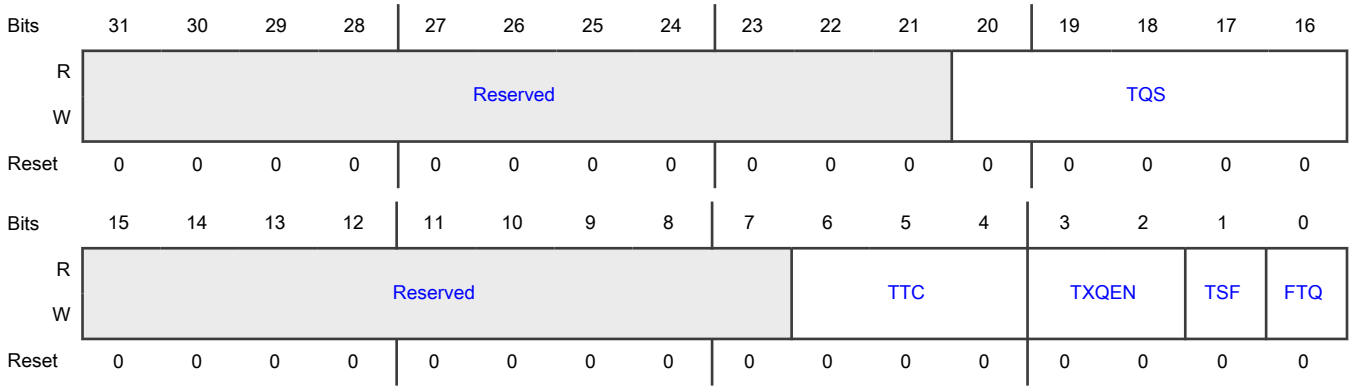
Offset

Register	Offset
MTL_TxQ1_Operation_Mode	D40h

Function

Establishes the transmit queue operating modes and commands.

Diagram



Fields

Field	Function
31-21 —	Reserved.
20-16 TQS	Transmit Queue Size Indicates the size of the allocated transmit queues in blocks of 256 bytes. This field is read-write only if the number of transit queues is more than one, the reset value is 0x0 and indicates size of 256 bytes. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. You must program TQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes. In general, the size of the queue = (TQS+1)*256 bytes. When the number of transit queues is one, the field is read-only and the reset value reflects the configured TX FIFO size in blocks of 256 bytes. The field width depends on the transit memory size selected in your configuration. For example, if the memory size is 2048, the field width is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
15-7 —	Reserved.
6-4 TTC	Transmit Threshold Control Controls the threshold level of the MTL Tx Queue. The transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, it also transmit full packets with length less than the threshold. These fields are used only when the MTL_TxQ1_Operation_Mode[TSF] resets. 000b - 32 001b - 64 010b - 96 011b - 128 100b - 192 101b - 256

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - 384 111b - 512
3-2 TXQEN	Transmit Queue Enable Enables or disables the transmit queue 0. 2'b00 - Not enabled 2'b01 - Enable in AV mode 2'b10 - Enabled 2'b11 - Reserved **NOTE** All the queues disables by default, in multiple transit queues configuration. You must program this field to enable the transit queue. 00b - Not enabled 01b - Enable in AV mode (Reserved in non-AV) 10b - Enabled 11b - Reserved
1 TSF	Transmit Store and Forward Indicates whether the transmit store and forward is enabled. When this field is 1, it indicates that the transmission starts when full packet resides in the MTL transit queue and the TTC values specified in MTL_TxQ1_Operation_Mode[TTC] are ignored. This field must change only when the transmission stops. 0b - Transmit Store and Forward is disabled 1b - Transmit Store and Forward is enabled
0 FTQ	Flush Transmit Queue Indicates whether the flush transmit queue is enabled. When this field is 1, it indicates that the transmit queue controller logic resets to its default values. Therefore, all the data in the transit queue is lost or flushed. This field resets internally when the flushing operation completes. Until this field becomes 0, you must not write to MTL_TxQ1_Operation_Mode. The data which the MAC transmitter has already accepted is not flushed. It is scheduled for transmission and results in an underflow and runt packet transmission. **NOTE** The flush operation completes only when the transit queue is empty and the application has accepted the pending transit status of all transmitted packets. To complete this flush operation, the PHY Tx clock (clk_tx_i) must be active. Access restriction apply to this field. Writing 1 sets this field and it clears automatically. Writing 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled

72.18.226 MTL Tx Queue 1 Underflow (MTL_TxQ1_Underflow)

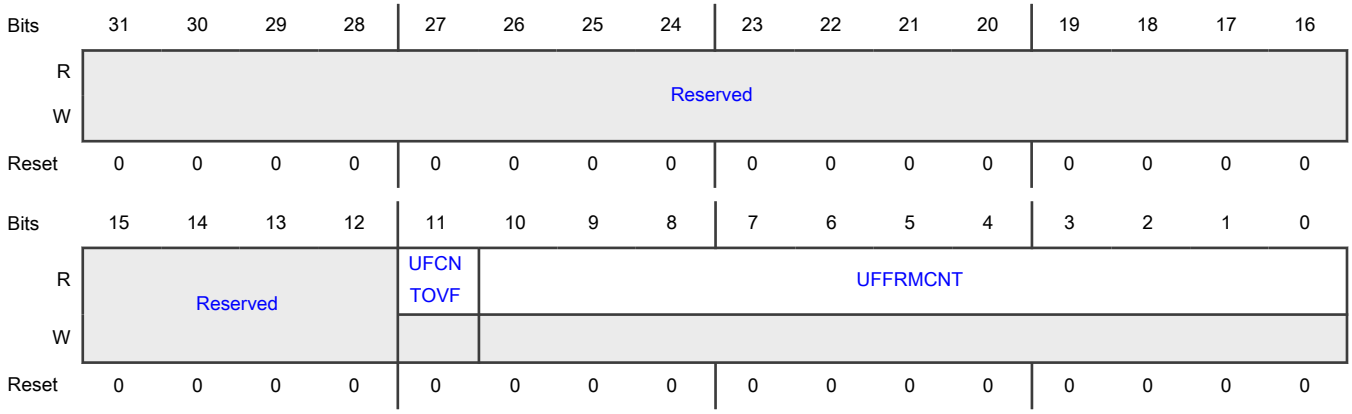
Offset

Register	Offset
MTL_TxQ1_Underflow	D44h

Function

Contains the counter for packets aborted because of transmit queue underflow and packets missed because of receive queue packet flush.

Diagram



Fields

Field	Function
31-12 —	Reserved.
11 UFCNTOVF	Overflow Bit for Underflow Packet Counter Indicates whether the overflow is detected for underflow packet counter. When this field is 1, it indicates that every time the transit queue underflow packet counter field overflows, that is, it has crossed the maximum count. In such case, the overflow packet counter resets to all-zeros and this field indicates that the rollover have occurred.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
10-0 UFFRMCNT	Underflow Packet Counter Indicates the number of packets which the controller abort because of transit queue underflow. This counter increments each time the MAC aborts outgoing packet because of underflow. The counter clears when this register is read with mci_be_i[0] at 1'b1. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

72.18.227 MTL Tx Queue 1 Debug (MTL_TxQ1_Debug)

Offset

Register	Offset
MTL_TxQ1_Debug	D48h

Function

Provides the debug status of various blocks related to the transmit queue.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved									STXSTSF			Reserved	PTXQ		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										TXSTSFS...	TXQSTS	TWCSTS	TRCSTS		TXQPAUS...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-23	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
22-20 STXSTSF	Number of Status Words in Tx Status FIFO of Queue Indicates the current number of status in the transit status FIFO of this queue. When MTL_Operation_Mode[DTXSTS] = 1, this field does not reflect the number of status words in transit status FIFO.
19 —	Reserved.
18-16 PTXQ	Number of Packets in the Transmit Queue Indicates the current number of packets in the transit queue. When MTL_Operation_Mode[DTXSTS] = 1, this field does not reflect the number of packets in the transmit queue.
15-6 —	Reserved.
5 TXSTSFSTS	MTL Tx Status FIFO Full Status Indicates whether the MTL transit status FIFO full status is detected. When this field is 1, it indicates that the MTL transit status FIFO is full. Therefore, the MTL cannot accept any more packets for transmission. 0b - Not detected 1b - Detected
4 TXQSTS	MTL Tx Queue Not Empty Status Indicates whether the MTL transit queue not empty status is detected. When this field is 1, it indicates that the MTL transit queue is not empty and some data is left for transmission. 0b - Not detected 1b - Detected
3 TWCSTS	MTL Tx Queue Write Controller Status Indicates whether the MTL transit queue write controller status is detected. When this field is 1, it indicates that the MTL transit queue write controller is active, and it is transferring the data to the transit queue. 0b - Not detected 1b - Detected
2-1 TRCSTS	MTL Tx Queue Read Controller Status Indicates the state of the transit queue read controller.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Idle state 01b - Read state (transferring data to the MAC transmitter) 10b - Waiting for pending transit status from the MAC transmitter 11b - Flushing the transit queue because of the packet abort request from the MAC
0 TXQPAUSED	Transmit Queue in Pause Indicates whether the transmit queue in pause status is detected. When this field is 1 and the receive flow control is enabled, it indicates that the transit queue is in the pause condition (in the full-duplex only mode) because of these scenarios: • Reception of the PFC packet for the priorities assigned to the transit queue when PFC is enabled. • Reception of 802.3x pause packet when PFC is disabled. 0b - Not detected 1b - Detected

72.18.228 MTL Tx Queue 1 ETS Control (MTL_TxQ1_ETS_Control)

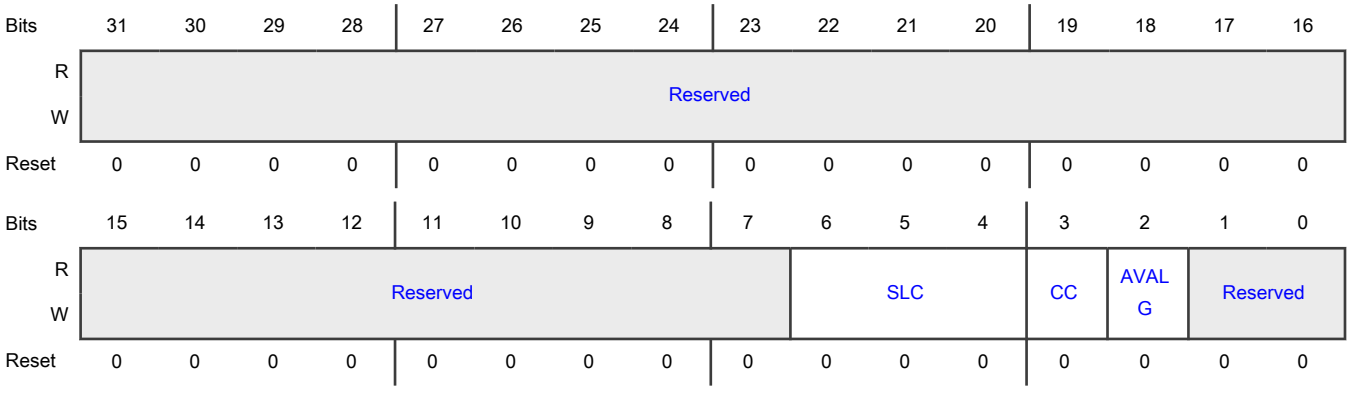
Offset

Register	Offset
MTL_TxQ1_ETS_Control	D50h

Function

Controls the enhanced transmission selection operation.

Diagram



Fields

Field	Function
31-7 —	Reserved.
6-4 SLC	Slot Count If the credit-based shaper algorithm is enabled, then you can program the number of slots (of duration programmed in DMA_CH(#i)_Slot_Interval register) over which the average transmitted bits per slot, provided in the MTL_TxQ(#i)_ETS_Status register, is computed for queue. The encoding is as follows: 000b - 1 slot 001b - 2 slots 010b - 4 slots 011b - 8 slots 100b - 16 slots 101b - Reserved
3 CC	Credit Control Indicates whether the credit control is enabled. When this field is 1, it indicates that the accumulated credit parameter in the credit-based shaper algorithm logic is not reset to zero, when there is positive credit and no packet to transmit in channel 1. The credit accumulates even when no packet is waiting in channel 1 and another channel is transmitting. When this field becomes 0, it indicates that the accumulated credit parameter in the credit-based shaper algorithm logic sets to zero, when there is positive credit and no packet to transmit in channel 1. No credit accumulates when no packet is waiting in channel 1 and other channel is transmitting. 0b - Disabled 1b - Enabled
2 AVALG	AV Algorithm Indicates whether the CBS algorithm is enabled. If you program queue 1 for AV, this field configures the scheduling algorithm for this queue. When this field is 1, it indicates that the credit based shaper algorithm (CBS) is selected for queue 1 traffic. If this field becomes 0, strict priority is selected. 0b - Disabled 1b - Enabled
1-0 —	Reserved.

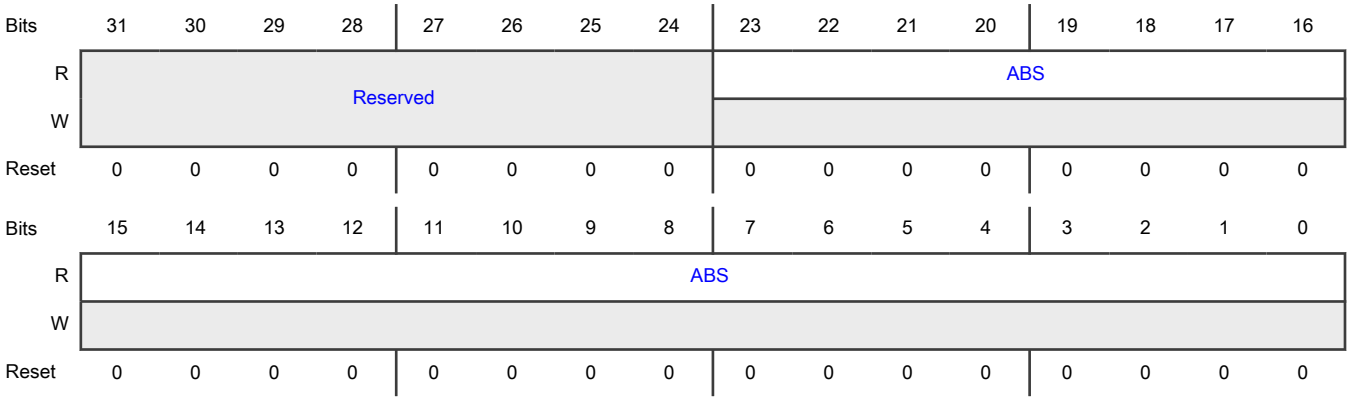
72.18.229 MTL Tx Queue 1 ETS Status (MTL_TxQ1_ETS_Status)

Offset

Register	Offset
MTL_TxQ1_ETS_Status	D54h

Function
Provides the average traffic transmitted in queue 1.

Diagram



Fields

Field	Function
31-24 —	Reserved.
23-0 ABS	Average Bits per Slot Contains the average transmitted bits per slot. Computes this field over number of slots programmed in the MTL_TxQ(#i)_ETS_CONTROL register's SLC field of, if you enables AV operation. This field's maximum value is 0x6_4000 in 100 Mbit/s, 0x3E_8000 in 1000 Mbit/s and 9C_4000 in 2500 Mbit/s mode respectively. Computes this field over every 10 million bit times slot (4 ms in 2500 Mbit/s; 10 ms in 1000 Mbit/s; 100 ms in 100 Mbit/s) when you enables DCB operation for queue. The maximum value is 0x989680.

72.18.230 MTL Tx Queue 1 Quantum Weight (MTL_TxQ1_Quantum_Weight)

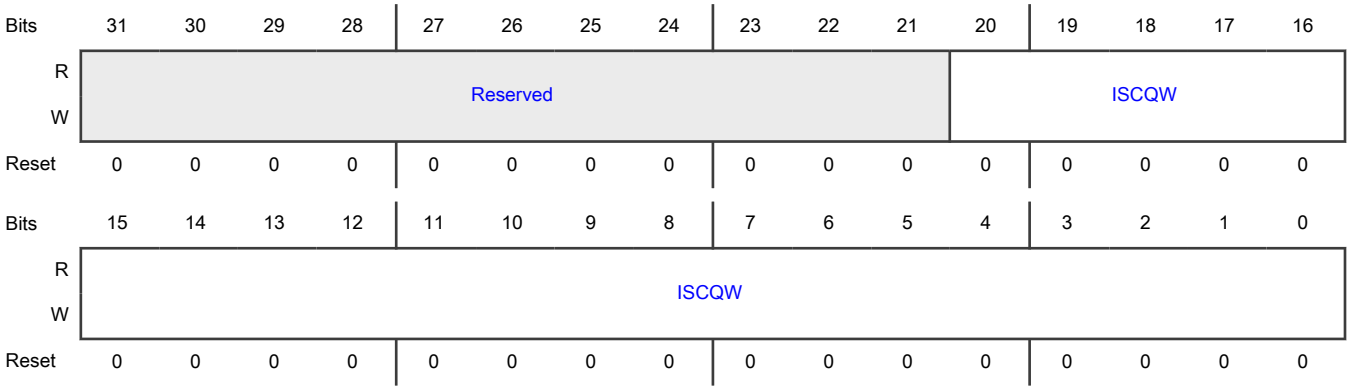
Offset

Register	Offset
MTL_TxQ1_Quantum_Weight	D58h

Function

Provides the average traffic transmitted in queue 1.

Diagram



Fields

Field	Function
31-21 —	Reserved.
20-0 ISCQW	idleSlopeCredit, Quantum or Weights idleSlopeCredit Contains the idleSlopeCredit value required for the credit-based shaper algorithm for queue 1, when you enables the AV feature. This is the rate of change of credit in bits per cycle (40 ns for 100 Mbit/s; 8 ns for 1000 Mbit/s; 3.2 ns for 2500 Mbit/s) when the credit is increasing. You must program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbit/s mode and 0x1000 in 100 Mbit/s mode. Bits[20:14] must be written to zero. Quantum Contains the quantum value in bytes to be added to credit during every queue scanning cycle, when you enables the DCB operation with DWRR algorithm for queue 1 traffic. The maximum value is 0x1312D0 bytes. Weights Contains the weight for this queue, when you enables the DCB operation with WFQ algorithm for queue 1 traffic. The maximum value is 0x3FFF where weight of 0 indicates 100% bandwidth. Bits[20:14] must be written to zero. Contains the weight for this queue, when you enables the DCB operation or generic queuing operation with WRR algorithm for queue 1 traffic. The maximum value is 0x64. Bits [20:7] must be written to zero.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE In multiple Queue configuration you must program this field in respective per queue register to some non-zero value when multiple queues are enabled or single queue other than Q0 is enabled. You must not program this field when only Q0 is enabled. In general, you must program a non-zero value on both receive and transmit when WRR algorithm is selected. In receive, the register is MTL_Operation_Mode register. **NOTE** For WFQ algorithm, higher the programmed weights lesser the bandwidth allocated for that transmit queue. The finish time is not a function of particular packet alone but it is according to the formula: (previous_finish_time of particular Transmit Queue + (weights*packet_size)) **NOTE** The programmed weights do not correspond to the number of packets but the fraction of bandwidth or time allocated for particular queue w.r.t. total BW or time.

72.18.231 MTL Tx Queue 1 Sendslope Credit (MTL_TxQ1_SendSlopeCredit)

Offset

Register	Offset
MTL_TxQ1_SendSlopeCredit	D5Ch

Function

Contains the sendSlope credit value required for the credit-based shaper algorithm for the queue.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		SSC													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-14 —	Reserved.
13-0 SSC	sendSlopeCredit Value Contains the sendSlopeCredit value required for credit-based shaper algorithm for Queue 1, when you enables the AV operation. This is the rate of change of credit in bits per cycle (40 ns, 8 ns and 3.2 ns for 100 Mbit/s, 1000 Mbit/s and 2500 Mbit/s respectively) when the credit is decreasing. You must program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbit/s mode and 0x1000 in 100 Mbit/s mode. You must program this fiels with absolute sendSlopeCredit value. The credit-based shaper logic subtracts it from the accumulated credit when channel 1 is selected for transmission.

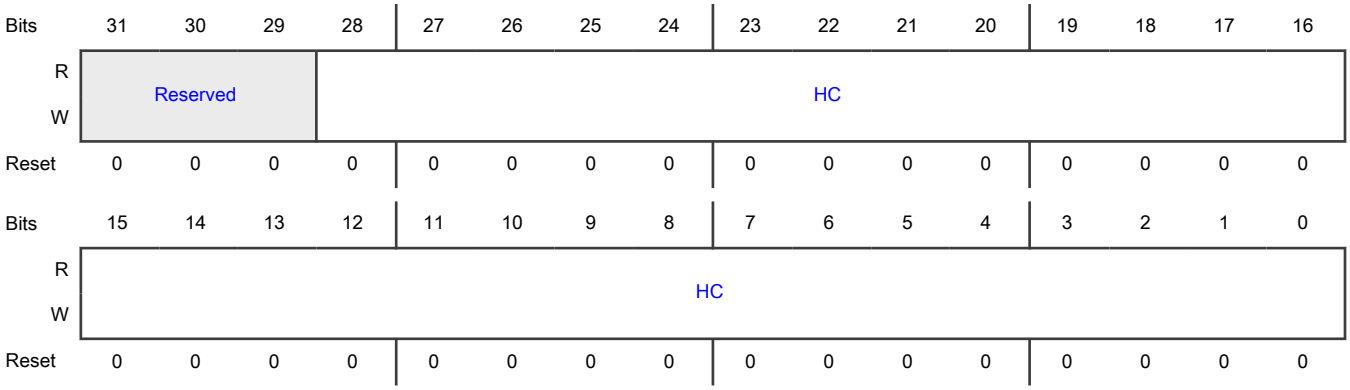
72.18.232 MTL Tx Queue 1 HiCredit (MTL_TxQ1_HiCredit)

Offset

Register	Offset
MTL_TxQ1_HiCredit	D60h

Function
Contains the hiCredit value required for the credit-based shaper algorithm for the queue.

Diagram



Fields

Field	Function
31-29 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-0 HC	hiCredit Value Contains the hiCredit value required for the credit-based shaper algorithm, when you enables the AV feature. This is the maximum value that you can accumulate in the credit parameter. This is specified in bits scaled by 1,024. The maximum value is maxInterferenceSize, that is, best-effort maximum packet size (16,384 bytes or 131,072 bits). The specified value is 131,072 * 1,024 = 134,217,728 or 0x0800_0000.

72.18.233 MTL Tx Queue 1 LoCredit (MTL_TxQ1_LoCredit)

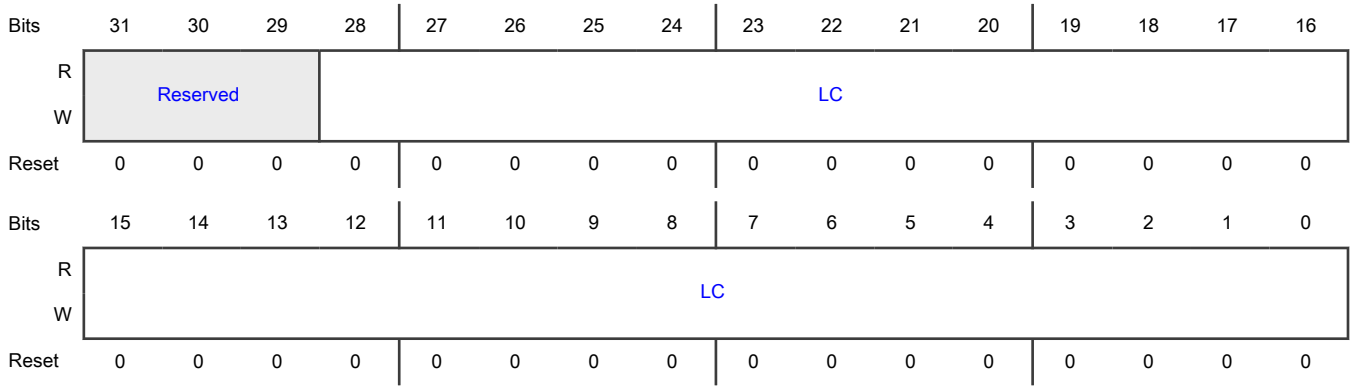
Offset

Register	Offset
MTL_TxQ1_LoCredit	D64h

Function

Contains the loCredit value required for the credit-based shaper algorithm for the queue.

Diagram



Fields

Field	Function
31-29 —	Reserved.
28-0 LC	loCredit Value Contains the loCredit value required for the credit-based shaper algorithm, when you enables the AV operation. This is the minimum value that you can accumulate in the credit parameter. This is specified in bits scaled by 1,024. The maximum value you can program corresponds to twice the maxFrameSize this queue transmits. If the maxFrameSize is 8192 bytes, then (8192*2) * 8 * 1024 = 134,217,728 or 0x0800_0000. The programmed value is 2's complement of the value, that is, 0x1800_0000 because it is a negative value.

72.18.234 MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status)

Offset

Register	Offset
MTL_Q1_Interrupt_Control_Status	D6Ch

Function
Contains the interrupt enable and status bits for the queue 1 interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved							RXOIE	Reserved							RXOVFIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						ABPSIE	TXUIE	Reserved						ABPSIS	TXUNFIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved.
24 RXOIE	Receive Queue Overflow Interrupt Enable Enables or disables the receive queue overflow interrupt. When this field is 1, it enables the receive queue overflow interrupt. When this field becomes 0, it disables the receive queue overflow interrupt. 0b - Disable 1b - Enable
23-17 —	Reserved.
16 RXOVFIS	Receive Queue Overflow Interrupt Status Indicates whether the status of receive queue overflow interrupt is detected. Indicates that the receive queue had an overflow when it receives the packet. If you transfers a partial packet to the application, the overflow status sets in RDES3[21]. This field is 0 when the application writes 1 to it.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
15-10 —	Reserved.
9 ABPSIE	Average Bits Per Slot Interrupt Enable Indicates whether an average bits per slot interrupt is enabled. When this field is 1, it indicates that the MAC asserts the sbd_intr_o or mci_intr_o interrupt when you update the average bits per slot status. When this field is 0, it indicates that the interrupt is not asserted for such an event. 0b - Disabled 1b - Enabled
8 TXUIE	Transmit Queue Underflow Interrupt Enable Enables or disables the transmit queue underflow interrupt. When this field is 1, it enables the transmit queue underflow interrupt. When this field becomes 0, it disables the transmit queue underflow interrupt. 0b - Disabled 1b - Enabled
7-2 —	Reserved.
1 ABPSIS	Average Bits Per Slot Interrupt Status Indicates whether the average bits per slot interrupt status is detected. When this field is 1, it indicates that the MAC has updated the ABS value. This field is 0 when the application writes 1 to it. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
0 TXUNFIS	Transmit Queue Underflow Interrupt Status Indicates whether the transmit queue underflow interrupt status is detected. Indicates that the transmit queue had an underflow when you transmits the packet. Suspends the transmission and sets an underflow error TDES3[2]. This field is 0 when the application writes 1 to this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writting 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected

72.18.235 MTL Rx Queue 1 Operation Mode (MTL_RxQ1_Operation_Mode)

Offset

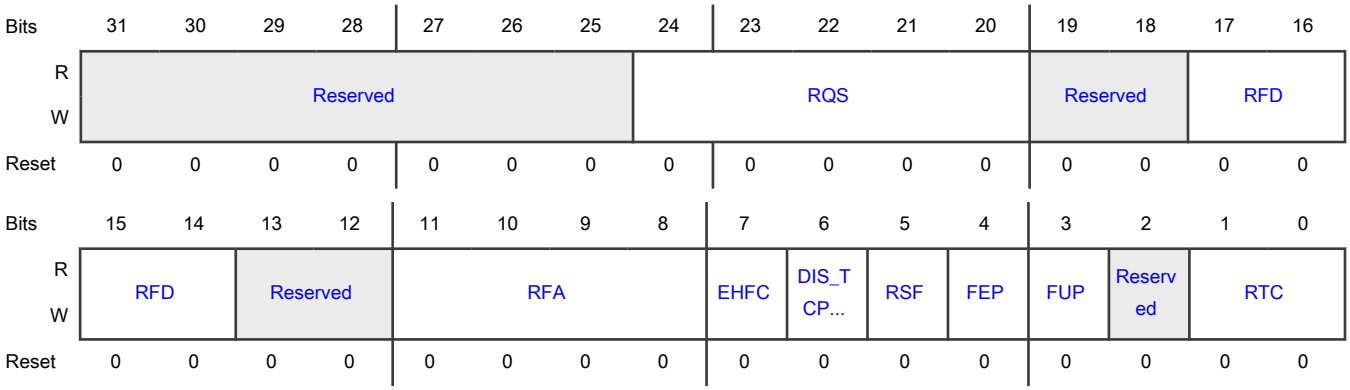
Register	Offset
MTL_RxQ1_Operation_Mode	D70h

Function

Establishes the receive queue operating modes and command.

The MTL_RxQ1_Operation_Mode[RFA] and MTL_RxQ1_Operation_Mode[RFD] are not backward compatible with the RFA and RFD fields of 4.00a release.

Diagram



Fields

Field	Function
31-25 —	Reserved.
24-20 RQS	Receive Queue Size Indicates the size of the allocated receive queues in blocks of 256 bytes. MTL_RxQ1_Operation_Mode[RQS] is read-write only if the number of receive queues is more than one, the

Table continues on the next page...

Table continued from the previous page...

Field	Function
	reset value is 0x0 and indicates 256 bytes size. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. You must program RQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes. In general, the size of the Queue = (RQS+1)*256 bytes. When the number of receive queues is one, the field is read-only and the reset value reflects the configured RX FIFO size in blocks of 256 bytes. The field width depends on the receive memory size selected in your configuration. For example, if the memory size is 2048, the field width is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
19-18 —	Reserved.
17-14 RFD	Threshold for Deactivating Flow Control (in half-duplex and full-duplex modes) Controls the threshold (fill-level of Rx queue) at which the flow control de-asserts after activation 0 - Full minus 1 KB, that is, FULL 1 KB 1 - Full minus 1.5 KB, that is, FULL 1.5 KB 2 - Full minus 2 KB, that is, FULL 2 KB 3 -Full minus 2.5 KB, that is, FULL 2.5 KB ... 14 - Full minus 8 KB, that is, FULL 8 KB 15 - Full minus 8.5 KB, that is, FULL 8.5 KB The de-assertion is effective only when flow control asserts. **NOTE** Program the value in such a way that the threshold is a positive number. When MTL_RxQ1_Operation_Mode[EHFC] = 1, these values are applicable only when the receive queue size which MTL_RxQ1_Operation_Mode[RQS] determines, is equal to or greater than 4 KB. For a given queue size, the values ranges between 0 and the encoding for FULL minus (QSIZE - 0.5 KB) and all other values are illegal. Here the term FULL and QSIZE refers to the queue size which MTL_RxQ1_Operation_Mode[RQS] determines. The field width depends on RX FIFO size selected during the configuration. Remaining bits are reserved and read only.
13-12 —	Reserved.
11-8 RFA	Threshold for Activating Flow Control (in half-duplex and full-duplex Controls the threshold (fill-level of Rx queue) at which the flow control activates. See MTL_RxQ1_Operation_Mode[RFD], for more information on encoding for this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 EHFC	Enable Hardware Flow Control Enables or disables hardware flow control. If this field is 1, it enables the flow control signal operation, based on the receive queue's fill-level. When this field becomes 0, it disables the flow control operation. 0b - Disable 1b - Enable
6 DIS_TCP_EF	Disable Dropping of TCP or IP Checksum Error Packets Indicates whether the dropping of TCP or IP checksum error packets is enabled. When this field is 1, it indicates that the MAC does not drop the packets which only have the errors that the receive checksum offload engine detects. Such packets have errors only in the encapsulated payload. MAC receives an Ethernet packet in which there are no errors (including FCS error). When this field becomes 0, it indicates that all the error packets drop if MTL_RxQ1_Operation_Mode[FEP] resets. 0b - Enabled 1b - Disabled
5 RSF	Receive Queue Store and Forward Indicates whether the receive queue store and forward is enabled. When this field is 1, it indicates that the module reads a packet from the receive queue only after the complete packet is written to it and ignores MTL_RxQ1_Operation_Mode[RTC]. When this field becomes 0, it indicates that the receive queue operates in the Threshold (cut-through) mode, subject to the threshold which MTL_RxQ1_Operation_Mode[RTC] specifies. 0b - Disabled 1b - Enabled
4 FEP	Forward Error Packets Indicates whether the forward error packets are enabled. If this field becomes 0, it indicates that the receive queue drops the packets with an error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, the packet does not drops, if the start byte (write) pointer of a packet is already transferred to the read controller side in Threshold mode. When this field is 1, it indicates that all the packets except the runt error packets forward to the application or DMA. If MTL_RxQ1_Operation_Mode[RSF] is 1 and the receive queue overflows when a partial packet is written, the packet drops irrespective of the setting of this field. However, if MTL_RxQ1_Operation_Mode[RSF] resets and the receive queue overflows when a partial packet is written, you may forward a partial packet to the application or DMA. 0b - Disabled 1b - Enabled
3	Forward Undersized Good Packets

Table continues on the next page...

Table continued from the previous page...

Field	Function
FUP	Indicates whether the forward undersized good packets are enabled. When this field is 1, it indicates that the receive queue forwards the undersized good packets (packets with no error and length less than 64 bytes), including pad-bytes and CRC. When this field becomes 0, it indicates that the receive queue drops all the packets of less than 64 bytes, unless a packet is already transferred because of the lower value of Rx Threshold, for example, RTC = 01. 0b - Disabled 1b - Enabled
2 —	Reserved.
1-0 RTC	Receive Queue Threshold Control Controls the threshold level of the MTL Rx queue (in bytes): The received packet is transferred to the application or DMA when the packet size within the MTL receive queue is larger than the threshold. Also, the full packets with length less than the threshold are automatically transferred. This field is valid only when MTL_RxQ1_Operation_Mode[RSF] = 0. This field is ignored when MTL_RxQ1_Operation_Mode[RSF] = 1. 00b - 64 01b - 32 10b - 96 11b - 128

72.18.236 MTL Rx Queue 1 Missed Packet Overflow Counter (MTL_RxQ1_Missed_Packet_Overflow_Cnt)

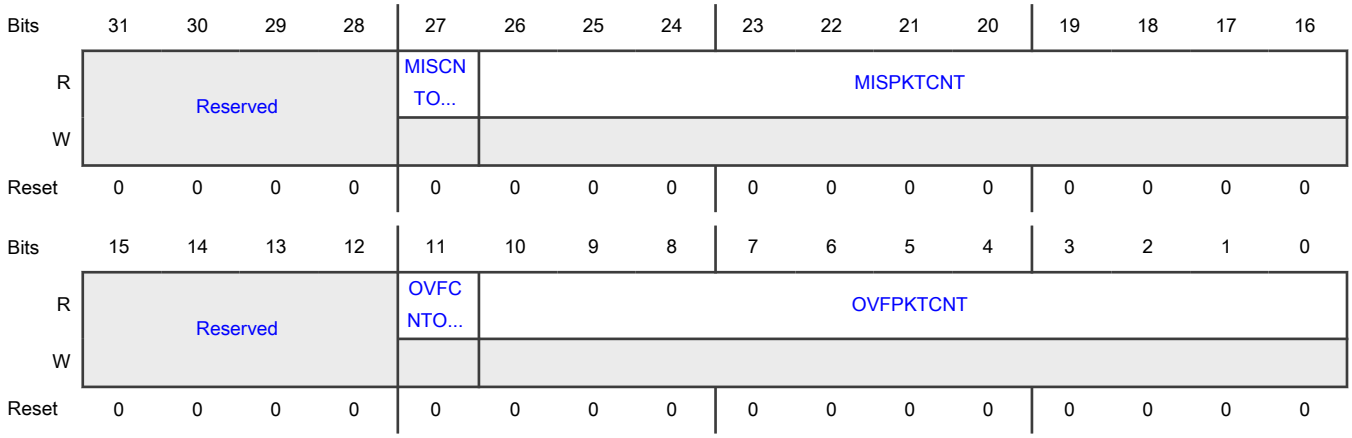
Offset

Register	Offset
MTL_RxQ1_Missed_Packet_Overflow_Cnt	D74h

Function

Contains the counter for packets missed because of receive queue packet flush and packets discarded because of receive queue overflow.

Diagram



Fields

Field	Function
31-28 —	Reserved.
27 MISCNTOVF	Missed Packet Counter Overflow Bit Indicates whether the missed packet counter overflow is detected. When this field is 1, it indicates that the receive queue missed packet counter has crossed the maximum limit. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
26-16 MISPKTCNT	Missed Packet Counter Indicates the number of packets the module has missed because the application asserted ari_pkt_flush_i[] for this queue. This counter resets when this register is read with mci_be_i[0] at 1b1. This counter increases by 1 when the DMA discards the packet because of buffer unavailability. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.
15-12 —	Reserved.
11 OVFCNTOVF	Overflow Counter Overflow Bit Indicates whether the counter overflow status is detected. When this field is 1, it indicates that the receive queue overflow packet counter field has crossed the maximum limit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
10-0 OVFPKTCNT	Overflow Packet Counter Indicates the number of packets which the module discards because of receive queue overflow. This counter increments each time the module discards an incoming packet because of overflow. It resets when this register is read with mci_be_i[0] at 1'b1. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

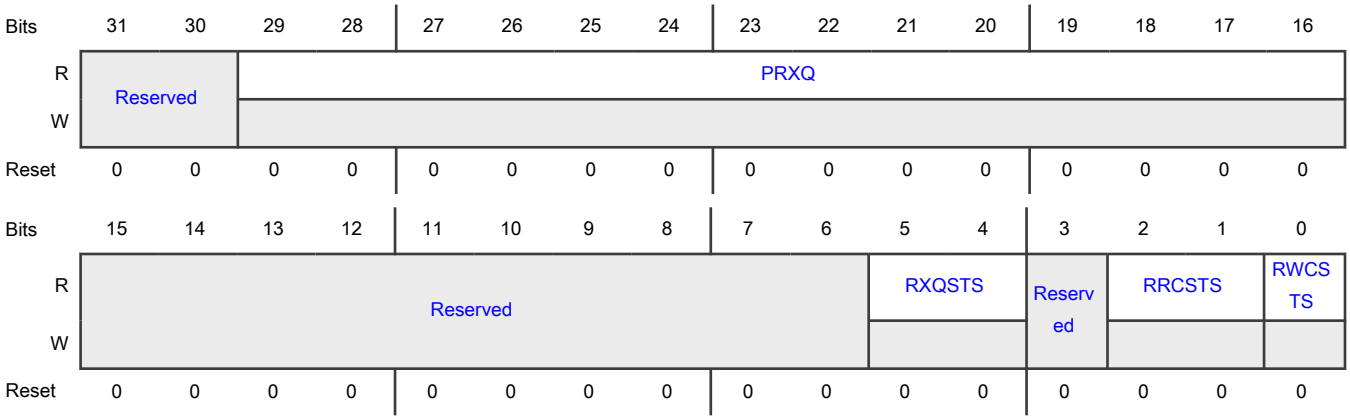
72.18.237 MTL Rx Queue 1 Debug (MTL_RxQ1_Debug)

Offset

Register	Offset
MTL_RxQ1_Debug	D78h

Function
Provides the debug status of various blocks related to the receive queue.

Diagram



Fields

Field	Function
31-30	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
29-16 PRXQ	Number of Packets in Receive Queue Indicates the current number of packets in the receive queue. The theoretical maximum field value is 256KB/16B = 16K packets, that is, Max_Queue_Size/Min_Packet_Size.
15-6 —	Reserved.
5-4 RXQSTS	MTL Rx Queue Fill-Level Status Provides the status of the receive queue fill-level. 00b - Rx Queue empty 01b - Rx Queue fill-level below flow-control deactivate threshold 10b - Rx Queue fill-level above flow-control activate threshold 11b - Rx Queue full
3 —	Reserved.
2-1 RRCSTS	MTL Rx Queue Read Controller State Provides the state of the receive queue read controller. 00b - Idle state 01b - Reading packet data 10b - Reading packet status (or timestamp) 11b - Flushing the packet data and status
0 RWCSTS	MTL Rx Queue Write Controller Active Status Indicates whether the MTL receive queue write controller active status is detected. When this field is 1, it indicates that the MTL receive queue write controller is active, and it is transferring a received packet to the receive queue. 0b - Not detected 1b - Detected

72.18.238 MTL Rx Queue 1 Control (MTL_RxQ1_Control)

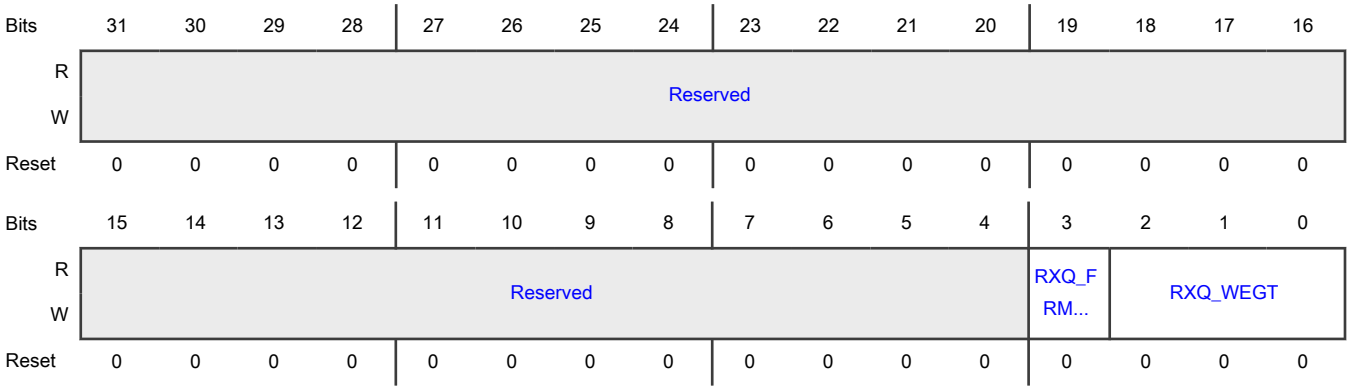
Offset

Register	Offset
MTL_RxQ1_Control	D7Ch

Function

Controls the receive arbitration and passing of received packets to the application.

Diagram



Fields

Field	Function
31-4 —	Reserved.
3 RXQ_FRM_ARBIT	Receive Queue Packet Arbitration Indicates whether the receive queue packet arbitration is enabled. When this field is 1, it indicates that the module drives the packet data to the ARI interface such that the entire packet data of currently-selected queue transmits before switching to other queue. When this field becomes 0, it indicates that the module drives the packet data to the ARI interface such that the following amount of data of currently-selected queue transmits before switching to other queue: • PBL amount of data (indicated by ari_qN_pbl_i[]), or • Complete data of a packet The status and the timestamp are not a part of the PBL data. Therefore, the module drives the complete status (including timestamp status) during the packet's first PBL request (in store-and-forward mode) or the last PBL request (in Threshold mode). 0b - Disabled 1b - Enabled
2-0 RXQ_WEGT	Receive Queue Weight Indicates the weight assigned to the receive queue 0. Program this field with one value less than the required weight, that is reset value of 0 indicates weight of 1, value of 1 indicates weight of 2, and so on. The weight is used as the number of continuous PBL or packets requests (depending on the RXQ_FRM_ARBIT) allocated to the queue in one arbitration cycle. Note: The change in value of RXQ_WEGT takes effect only after the completion of current service round or when there is change from RAA=SP to RAA=WSP algorithm. This approach is taken so that there is smooth transition. For the RXQ_WEGT value to take effect at the start, the MTL_RxQ(#i)_Control registers must be programmed before the MTL_Operation_Mode register.

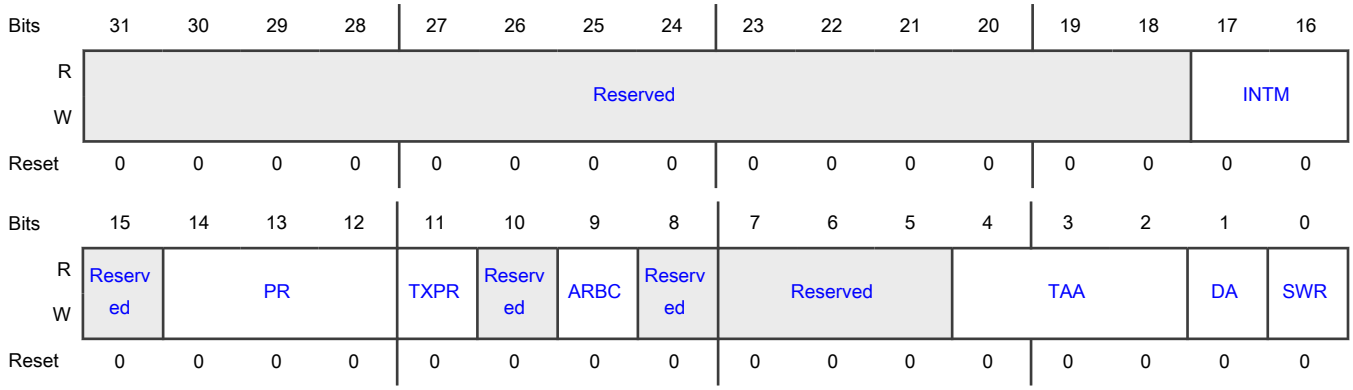
72.18.239 DMA Mode (DMA_Mode)

Offset

Register	Offset
DMA_Mode	1000h

Function
Establishes the bus operating modes for the DMA.

Diagram



Fields

Field	Function
31-18 —	Reserved.
17-16 INTM	Interrupt Mode Defines the interrupt mode of module. The behavior of the following outputs changes depending on these settings: sbd_perch_tx_intr_o[] (Transmit per channel interrupt) sbd_perch_rx_intr_o[] (Receive per channel interrupt) sbd_intr_o (Common interrupt) It also changes the behavior of the RI/TI bits in DMA_CH0_Status. 00 - sbd_perch_* are pulse signals for each TX/RX packet transfer completion events (irrespective of whether corresponding interrupts are enabled) for which IOC bits are enabled in descriptor. sbd_intr_o is also asserted when corresponding interrupts are enabled and cleared only when software clears the corresponding RI/TI status bits. 01 - sbd_perch_* are level signals asserted on TX/RX packet transfer completion event when corresponding interrupts are enabled and de-asserted when the software clears the corresponding RI/TI status bits. The sbd_intr_o is not asserted for these TX/RX packet transfer completion events.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10 - sbd_perch_* are level signals asserted on TX/RX packet transfer completion event when corresponding interrupts are enabled and de-asserted when the software clears the corresponding RI/TI status bits. However, the signal is asserted again if the same event occurred again before it was cleared. The sbd_intr_o is not asserted for these TX/RX packet transfer completion events. 11 - Reserved See table "Transfer Complete Interrupt Behavior" for more information. 00b - See above description 01b - See above description 10b - See above description 11b - Reserved
15 —	Reserved.
14-12 PR	Priority Ratio Controls the priority ratio in weighted round-robin arbitration between the Rx DMA and Tx DMA. These fields are valid only when DMA_Mode[DA] resets. The priority ratio is Rx:Tx or Tx:Rx depending on whether DMA_Mode[TXPR] = 1 or it resets. 000b - The priority ratio is 1:1 001b - The priority ratio is 2:1 010b - The priority ratio is 3:1 011b - The priority ratio is 4:1 100b - The priority ratio is 5:1 101b - The priority ratio is 6:1 110b - The priority ratio is 7:1 111b - The priority ratio is 8:1
11 TXPR	Transmit Priority Indicates whether the transmit priority is enabled. When this field is 1, it indicates that the Tx DMA has higher priority than the Rx DMA during arbitration for the system-side bus. 0b - Disabled 1b - Enabled
10 —	Reserved.
9 ARBC	Is reserved for NXP internal use. This field must always be 0 unless instructed by NXP. Writing 1 to this field may cause unexpected behavior.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - NXP reserved field disabled 1b - NXP reserved field enabled up on NXP request
8 —	Reserved.
7-5 —	Reserved.
4-2 TAA	Transmit Arbitration Algorithm Selects the arbitration algorithm for the transmit side when you select multiple transit DMAs. 000b - Fixed priority (Channel 0 has the lowest priority and the last channel has the highest priority) 001b - Weighted Strict Priority (WSP) 010b - Weighted Round-Robin (WRR) 011b - Reserved (for 3'b011 to 3'b111)
1 DA	DMA Tx or Rx Arbitration Scheme Specifies the arbitration scheme between the transmit and receive paths of all channels. 0 - Weighted round-robin with Rx:Tx or Tx:Rx The priority between the paths is according to the priority specified in bits[14:12] and the priority weight is specified in DMA_Mode[TXPR]. 1 - Fixed Priority The transit path has priority over the receive path when DMA_Mode[TXPR] = 1. Otherwise, the receive path has priority over the transit path. 0b - Weighted Round-Robin with Rx:Tx or Tx:Rx 1b - Fixed Priority
0 SWR	Software Reset Indicates whether the software reset is enabled. When this field is 1, it indicates that the MAC and the DMA controller resets the logic and all internal registers of the DMA, MTL, and MAC. This field automatically clears after the reset operation completes in all module clock domains. A value of zero should be read in this field before reprogramming any module register. When this field is 1, it indicates that this field must read at least 4 CSR clock cycles. **NOTE** The reset operation completes only when all resets in all active clock domains de-assert. Therefore, to complete software reset, it is essential that all PHY input clocks (applicable for the selected PHY interface) are present. The time to complete the software reset operation depends on the frequency of the slowest active clock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It clears automatically. Writing 1 sets this field and writting 0 has no effect. 0b - Disabled 1b - Enabled

72.18.240 DMA System Bus Mode (DMA_SysBus_Mode)

Offset

Register	Offset
DMA_SysBus_Mode	1004h

Function

Controls the behavior of the AHB or AXI master. It mainly controls burst splitting and the number of outstanding requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	Reserved	Reserved				Reserved		Reserved						Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RB	MB	Reserved	AAL	Reserved	Reserved	Reserved		Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	FB
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved.
30 —	Reserved.
29-26 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved.
23-18 —	Reserved.
17-16 —	Reserved.
15 RB	Rebuild INCRx Burst Indicates whether the rebuild INCRx burst is enabled. When this field is 1 and the AHB master gets SPLIT, RETRY, or Early Burst Termination (EBT) response, it indicates that the AHB master interface rebuilds the pending beats of any initiated burst transfer with INCRx and SINGLE transfers. By default, the AHB master interface rebuilds pending beats of an EBT with an unspecified (INCR) burst. 0b - Disabled 1b - Enabled
14 MB	Mixed Burst Indicates whether the mixed burst is enabled. When this field is 1 and DMA_SysBus_Mode[FB] is 0, it indicates that the AHB master performs an undefined bursts transfers (INCR) for burst length of 16 or more. The AHB master performs fixed burst transfers (INCRx and SINGLE), for burst length of 16 or less. 0b - Disabled 1b - Enabled
13 —	Reserved.
12 AAL	Address-Aligned Beats Indicates whether the address-aligned beats are enabled. When this field is 1, it indicates that the EQOS-AXI or EQOS-AHB master performs address-aligned burst transfers on read and write channels. 0b - Disabled 1b - Enabled
11 —	Reserved.
10 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-8 —	Reserved.
7 —	Reserved.
6 —	Reserved.
5 —	Reserved.
4 —	Reserved.
3 —	Reserved.
2 —	Reserved.
1 —	Reserved.
0 FB	Fixed Burst Length Indicates whether the fixed burst length is enabled. When this field is 1, it indicates that the AHB master initiates burst transfers of specified length (INCRx or SINGLE). When this field is 0, it indicates that the AHB master initiates transfers of unspecified length (INCR) or SINGLE transfers. 0b - Disabled 1b - Enabled

72.18.241 DMA Interrupt Status (DMA_Interrupt_Status)

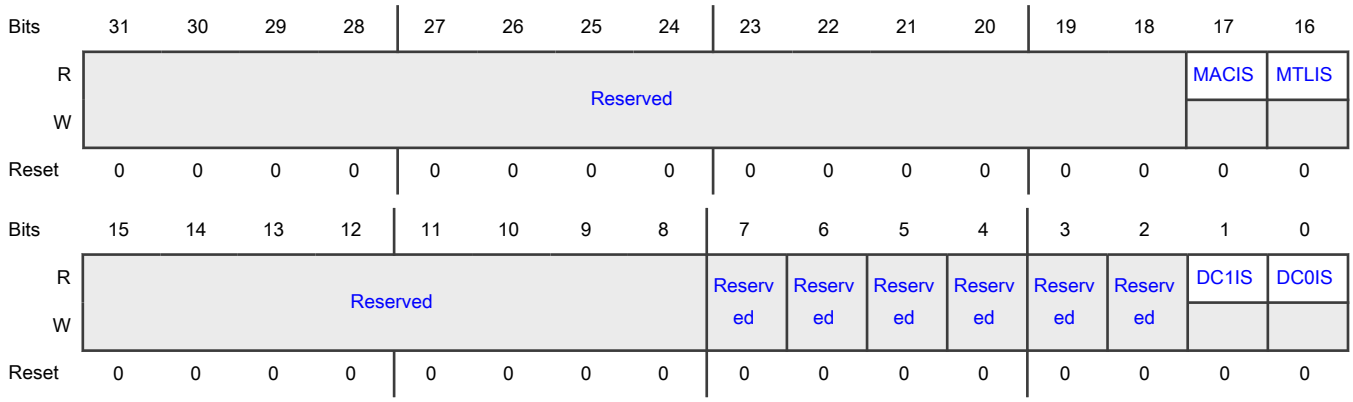
Offset

Register	Offset
DMA_Interrupt_Status	1008h

Function

The application reads this Interrupt Status register during interrupt service routine or polling to determine the interrupt status of DMA channels, MTL queues, and the MAC.

Diagram



Fields

Field	Function
31-18 —	Reserved.
17 MACIS	MAC Interrupt Status Indicates whether the MAC interrupt status is detected. You must read the corresponding register in the MAC to get the exact cause of the interrupt and clear its source, to reset this field to 1'b0. 0b - Not detected 1b - Detected
16 MTLIS	MTL Interrupt Status Indicates whether the MTL interrupt status is detected. You must read the corresponding register in the MTL to get the exact cause of the interrupt and clear its source, to reset this field to 1'b0. 0b - Not detected 1b - Detected
15-8 —	Reserved.
7 —	Reserved.
6	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5 —	Reserved.
4 —	Reserved.
3 —	Reserved.
2 —	Reserved.
1 DC1IS	DMA Channel 1 Interrupt Status Indicates whether the DMA channel 1 interrupt status is detected. You must read the corresponding register in DMA channel 1 to get the exact cause of the interrupt and clear its source, to reset this field to 1'b0. 0b - Not detected 1b - Detected
0 DC0IS	DMA Channel 0 Interrupt Status Indicates whether the DMA channel 0 interrupt status is detected. You must read the corresponding register in DMA channel 0 to get the exact cause of the interrupt and clear its source, to reset this field to 1'b0. 0b - Not detected 1b - Detected

72.18.242 DMA Debug Status 0 (DMA_Debug_Status0)

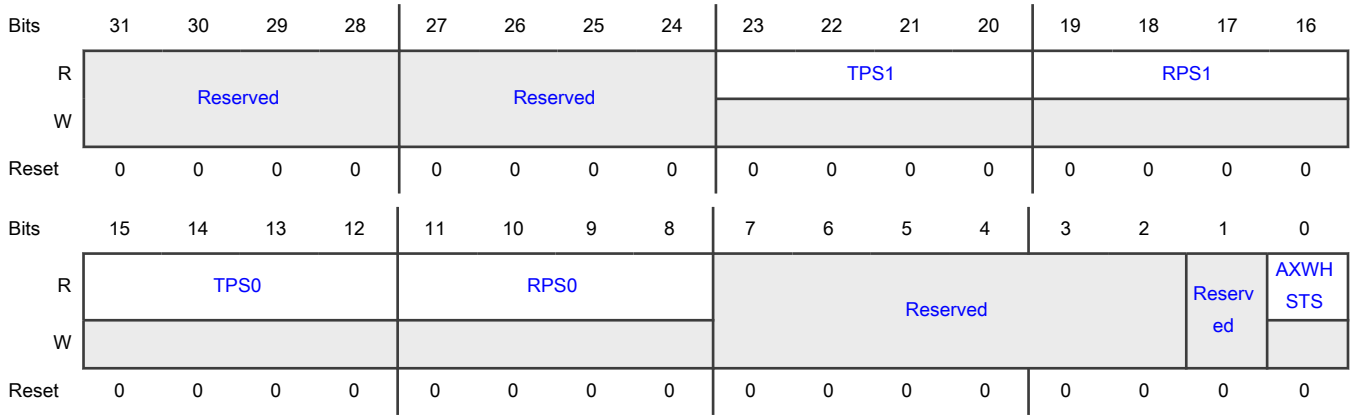
Offset

Register	Offset
DMA_Debug_Status0	100Ch

Function

Provides the receive and transmit process status for DMA Channel 0-Channel 2 for debugging purpose.

Diagram



Fields

Field	Function
31-28 —	Reserved.
27-24 —	Reserved.
23-20 TPS1	DMA Channel 1 Transmit Process State Indicates the transmit DMA FSM state for channel 1. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or stop transmit command issued) 0001b - Running (Fetching transmit transfer descriptor) 0010b - Running (Waiting for status) 0011b - Running (Reading data from system memory buffer and queuing it to the transmit buffer (Tx FIFO)) 0100b - Timestamp write state 0101b - Reserved for future use 0110b - Suspended (Transmit descriptor unavailable or transmit buffer underflow) 0111b - Running (Closing transmit descriptor)
19-16 RPS1	DMA Channel 1 Receive Process State Indicates the receive DMA FSM state for channel 1. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or Stop receive command issued) 0001b - Running (Fetching receive transfer descriptor) 0010b - Reserved for future use

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0011b - Running (Waiting for receive packet) 0100b - Suspended (Receive descriptor unavailable) 0101b - Running (Closing the receive descriptor) 0110b - Timestamp write state 0111b - Running (Transferring the received packet data from the receive buffer to the system memory)
15-12 TPS0	DMA Channel 0 Transmit Process State Indicates the transmit DMA FSM state for channel 0. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or stop transmit command issued) 0001b - Running (Fetching transmit transfer descriptor) 0010b - Running (Waiting for status) 0011b - Running (Reading data from system memory buffer and queuing it to the transmit buffer (Tx FIFO)) 0100b - Timestamp write state 0101b - Reserved for future use 0110b - Suspended (Transmit descriptor unavailable or transmit buffer underflow) 0111b - Running (Closing transmit descriptor)
11-8 RPS0	DMA Channel 0 Receive Process State Indicates the receive DMA FSM state for channel 0. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or stop receive command issued) 0001b - Running (Fetching receive transfer descriptor) 0010b - Reserved for future use 0011b - Running (Waiting for receive packet) 0100b - Suspended (Receive descriptor unavailable) 0101b - Running (Closing the receive descriptor) 0110b - Timestamp write state 0111b - Running (Transferring the received packet data from the receive buffer to the system memory)
7-2 —	Reserved.
1	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 AXWHSTS	AHB Master Status Indicates whether the AXI master write channel or AHB master status is detected. When this field is 1, it indicates that the AHB master FSMs are in the non-idle state. 0b - Not detected 1b - detected

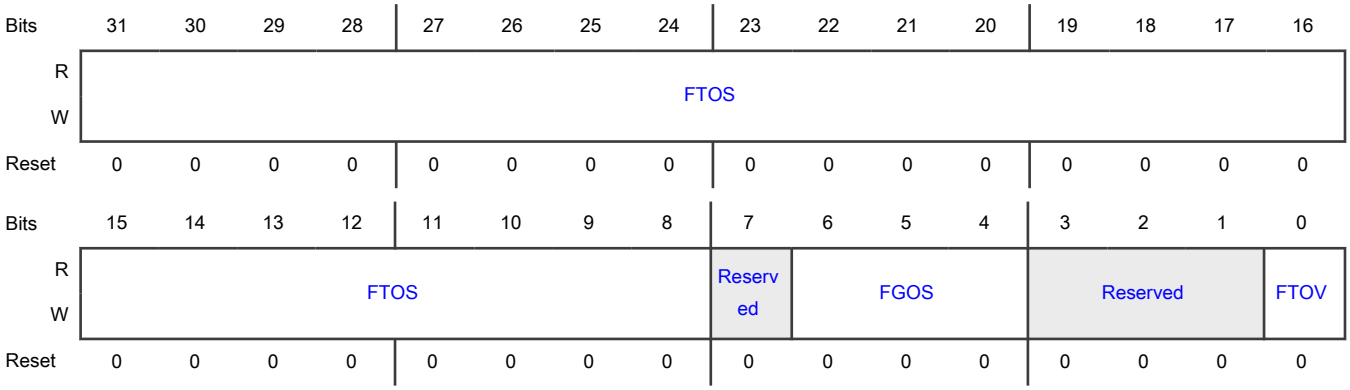
72.18.243 DMA TBS Control (DMA_TBS_CTRL)

Offset

Register	Offset
DMA_TBS_CTRL	1050h

Function
Controls the TBS attributes.

Diagram



Fields

Field	Function
31-8 FTOS	Fetch Time Offset Deduct the value in units of 256 nanoseconds, from the launch time to compute the fetch Time. Max value: 999,999,999 nanosecond, must be smaller than CTR-1 value when ESTM mode = 1, because this value is a modulo CTR value.
7	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6-4 FGOS	Fetch GSN Offset Deduct the number GSN slots from the launch GSN to compute the fetch GSN. The value is valid only when DMA_TBS_CTRL[FTOV] = 1.
3-1 —	Reserved.
0 FTOV	Fetch Time Offset Valid Indicates whether the fetch time offset is valid. When this field is 1, it indicates that DMA_TBS_CTRL[FTOS] is valid. When this field is not 1, it indicates that the fetch offset is not valid and the DMA engine can fetch the frames from host memory without any time restrictions. 0b - Invalid 1b - Valid

72.18.244 DMA Safety Interrupt Status (DMA_Safety_Interrupt_Status)

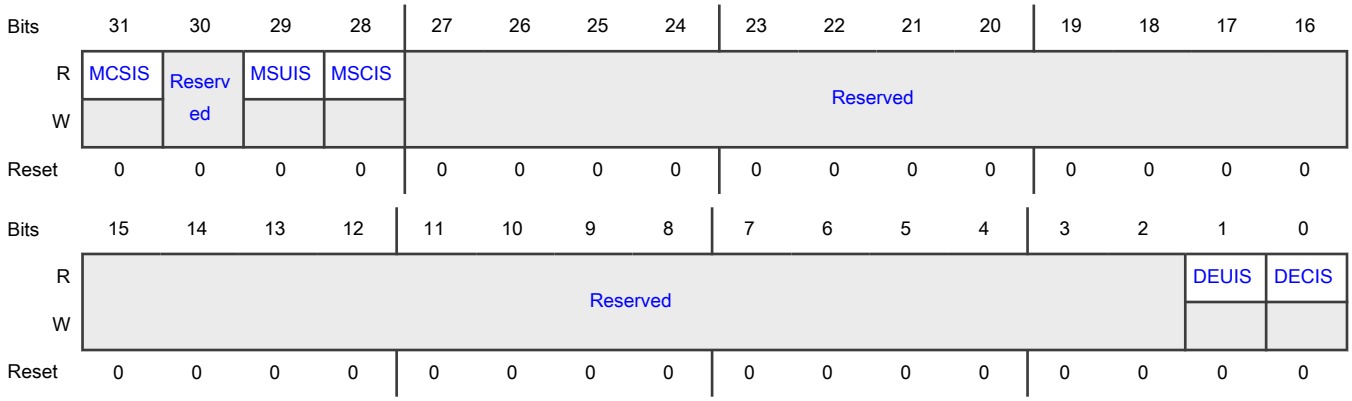
Offset

Register	Offset
DMA_Safety_Interrupt_Status	1080h

Function

Indicates summary (whether error occured in DMA/MTL/MAC and correctable/uncorrectable) of the automotive safety related error interrupts.

Diagram



Fields

Field	Function
31 MCSIS	MAC Safety Uncorrectable Interrupt Status Indicates whether the MAC safety uncorrectable interrupt status is detected. Indicates that an uncorrectable safety related interrupt is 1 in the MAC module. When this field is 1, it indicates that you must read the MAC_DPP_FSM_Interrupt_Status, to get the cause of the safety interrupt in MAC. 0b - Not detected 1b - Detected
30 —	Reserved.
29 MSUIS	MTL Safety Uncorrectable Error Interrupt Status Indicates whether the MTL safety uncorrectable error interrupt status is detected. Indicates an uncorrectable error interrupt event in MTL. To get exact cause of the interrupt you must read the MTL_Safety_Interrupt_Status. 0b - Not detected 1b - Detected
28 MSCIS	MTL Safety Correctable Error Interrupt Status Indicates whether the MTL safety correctable error interrupt status is detected. Indicates a correctable error interrupt event in MTL. To get the exact cause of the interrupt you must read the MTL_Safety_Interrupt_Status. 0b - Not detected 1b - Detected
27-2 —	Reserved.
1 DEUIS	DMA ECC Uncorrectable Error Interrupt Status Indicates whether the DMA ECC uncorrectable error interrupt status is deducted. Indicates an interrupt event in the DMA ECC safety feature. To get the exact cause of the interrupt the application must read the DMA_ECC_Interrupt_Status register. 0b - Not detected 1b - Detected
0 DECIS	DMA ECC Correctable Error Interrupt Status Indicates whether the DMA ECC correctable error interrupt status is detected. Indicates an interrupt event in the DMA ECC safety feature. To get the exact cause of the interrupt the application must read the DMA_ECC_Interrupt_Status register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected

72.18.245 DMA Channel 0 Control (DMA_CH0_Control)

Offset

Register	Offset
DMA_CH0_Control	1100h

Function

Specifies the MSS value for segmentation, length to skip between two descriptors, and also the features such as header splitting and 8xPBL mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved							Reserved	Reserved				DSL			Reserved	PBLx8
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved.
24 —	Reserved.
23-21 —	Reserved.
20-18	Descriptor Skip Length

Table continues on the next page...

Table continued from the previous page...

Field	Function
DSL	Specifies the Word, Dword, or Lword number (depending on the 32-bit, 64-bit, or 128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of the current descriptor to the start of the next descriptor. The DMA assumes the descriptor table as contiguous, when the DSL value is equal to zero.
17 —	Reserved.
16 PBLx8	8xPBL mode Indicates whether 8xPBL mode is enabled. When this field is 1, it indicates that the PBL value programmed in Bits[21:16] in DMA_CH(#i)_Tx_Control and Bits[21:16] in DMA_CH(#i)_Rx_Control is multiplied by eight times. Therefore, the DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. 0b - Disabled 1b - Enabled
15-14 —	Reserved.
13-0 —	Reserved.

72.18.246 DMA Channel Tx Control (DMA_CH0_Tx_Control)

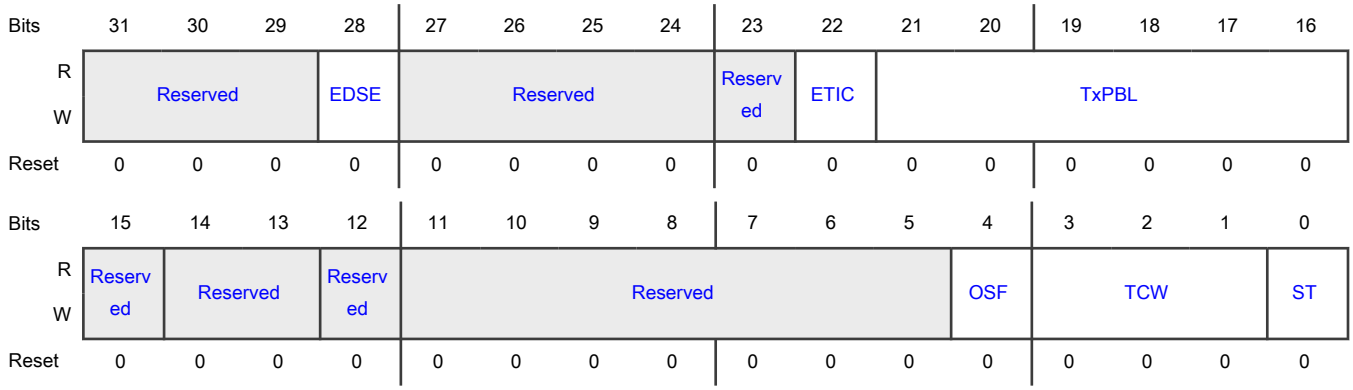
Offset

Register	Offset
DMA_CH0_Tx_Control	1104h

Function

Controls the transmit features such as PBL, TCP segmentation, and transmit channel weights.

Diagram



Fields

Field	Function
31-29 —	Reserved.
28 EDSE	Enhanced Descriptor Enable Indicates whether an enhanced descriptor is enabled. When this field is 1, it indicates that the corresponding channel uses 32 bytes enhanced descriptors for both normal and context descriptors. When this field becomes 0, it indicates that the corresponding channel uses the 16 bytes descriptors. 0b - Disabled 1b - Enabled
27-24 —	Reserved.
23 —	Reserved.
22 ETIC	Early Transmit Interrupt Control Indicates whether an early transmit interrupt is enabled. When this field is 1, it indicates that an early transmit interrupt (ETI) status = 1 after the data transfer from buffers of a transmit descriptor completes in which IOC bit (TDES2[31]) = 1. When this field becomes 1, it indicates that ETI = 1 only after a complete packet transfers to the MTL TX FIFO memory. 0b - Disabled 1b - Enabled
21-16 TxPBL	Transmit Programmable Burst Length

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates the maximum number of beats to transfer in one DMA block data transfer. DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of these values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. Perform these steps to transfer more than 32 beats: 1. Set DMA_CH0_Control[PBLx8]. 2. Set the TxPBL. **NOTE** The maximum value of TxPBL must be less than or equal to half the transit queue size (TQS field of MTL_TxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the transit queue has space to store at least another TxPBL worth of data while the MTL Tx queue controller transfers data to MAC. For example, in 64-bit data width configurations the total locations in transit queue of size 512 bytes is 64. You must program TxPBL and 8xPBL to less than or equal to 32.
15 —	Reserved.
14-13 —	Reserved.
12 —	Reserved.
11-5 —	Reserved.
4 OSF	Operate on Second Packet Indicates whether the operation on second packet is enabled. When this field is 1, it instructs DMA to process the second packet of the transmit data even before you receive the status for the first packet. 0b - Disabled 1b - Enabled
3-1 TCW	Transmit Channel Weight Indicates the weight assigned to the corresponding transmit channel. When reset completes, this field is 0 for all the channels by default, which results in equal weights to all channels.
0 ST	Start or Stop Transmission Command Indicates whether to start or stop the transmission command. When this field is 1, it indicates that the transmission is in the running state. DMA checks the transmit list at the current position for a packet to be transmitted.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DMA tries to acquire descriptor from either of these positions: • The current position in the list. This is the base address of the transmit list to which DMA_CH0_TxDesc_List_Address sets. • The position at which the transmission was previously stopped. The transmission enters the suspended state and DMA_CH0_Status[TBU] = 1, if DMA does not own the current descriptor. The start transmission command is effective only when the transmission stops. If the command is issued before setting DMA_CH0_TxDesc_List_Address, you cannot predict the DMA behavior. When this field becomes 0, it indicates that the transmission process is placed in the stopped state when the transmission of the current packet completes . The next descriptor position in the transmit list is saved, and it becomes the current position when you restart the transmission. To change the list address, you must program DMA_CH0_TxDesc_List_Address with a new value when this field becomes 0.You can consider the new value when this field is 1 again. The stop transmission command is effective only when the transmission of the current packet completes or the transmission is in the suspended state. 0b - Stop 1b - Start

72.18.247 DMA Channel Rx Control (DMA_CH0_Rx_Control)

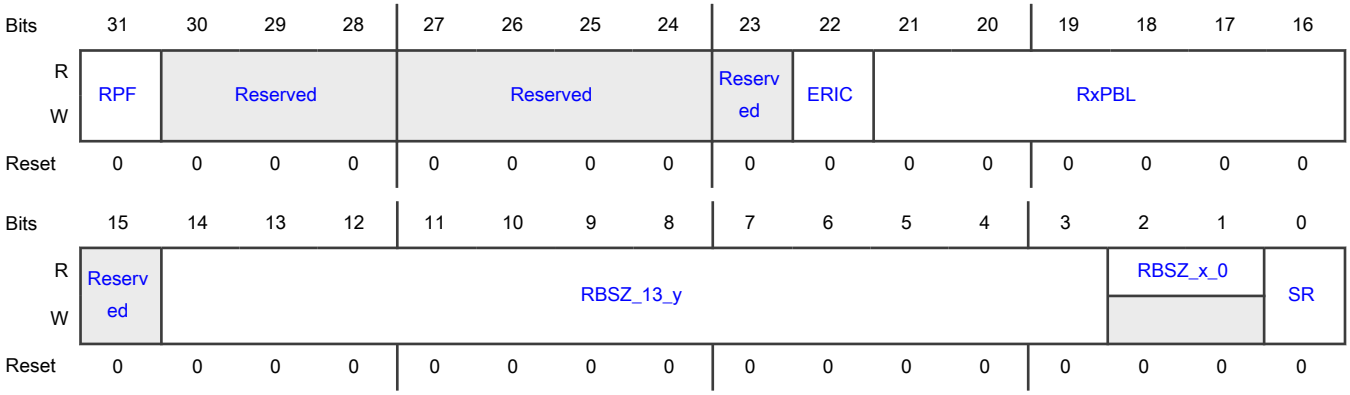
Offset

Register	Offset
DMA_CH0_Rx_Control	1108h

Function

Controls the receive features such as PBL, buffer size, and extended status.

Diagram



Fields

Field	Function
31 RPF	Rx Packet Flush Indicates whether the receive packet flush is enabled. When this field is 1, it indicates that the module automatically flushes the packet from the receive queues destined to this DMA receive channel, when it stops. When this field remains 1 and the software driver re-starts DMA, the packets residing in the receive queues that were received when this RxDMA stops and flushes out. The packets that MAC receives after the RxDMA re-starts, route to the RxDMA. The flushing takes place on the read side of the receive queue. When this field is 0, it indicates that the module do not flush the packet in the receive queue destined to this RxDMA channel when it is in STOP state. This may cause head-of-line blocking in the corresponding receive queue. **NOTE** The packet flow from a receive DMA channel to the application by writing 1 to this field stops, only when there is one-to-one mapping of receive queue to receive DMA channels. In Dynamic mapping mode, writing 1 to this field in any DMA_CH(#i)_Rx_Control register might flush packets from an unintended receive queues which are destined to the stopped receive DMA channel. 0b - Disabled 1b - Enabled
30-28 —	Reserved.
27-24 —	Reserved.
23 —	Reserved.
22 ERIC	Early Receive Interrupt Control Indicates whether an early receive interrupt is enabled. When this field is 1, it indicates that the early receive interrupt (ERI) status sets after every burst transfer of data from the Rx DMA to the buffer completes. When this field becomes 0, it indicates that ERI sets only after a complete buffer is filled by the RxDMA. 0b - Disabled 1b - Enabled
21-16 RxPBL	Receive Programmable Burst Length Indicates the maximum number of beats to be transferred in one DMA block data transfer. DMA always attempts max burst as specified in PBL every time it starts a burst transfer on the application bus. You can program PBL with any of these values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1. Set DMA_CH0_Control[PBLx8]. 2. Set the RxPBL. **NOTE** The maximum value of RxPBL must be less than or equal to half the receive queue size (RQS field of MTL_RxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the receive queue has space to store at least another Rx PBL worth of data while the receive DMA transfers a block of data. For example, in 64-bit data width configurations the total locations in receive queue of size 512 bytes is 64. You must program RxPBL and 8xPBL to less than or equal to 32.
15 —	Reserved.
14-3 RBSZ_13_y	Receive Buffer size High RBSZ[13:0] splits into two fields, higher field is RBSZ_13_y and lower field is RBSZ_x_0. RBSZ[13:0] field indicates the receive buffers size specified in bytes. The maximum buffer size is limited to 16K bytes. The buffer size is applicable to payload buffers when split headers are enabled. **NOTE** The buffer size must be a multiple of 4, 8, or 16 depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively). This is required even if the buffer address pointer value is not aligned with data bus width. Hence the lower RBSZ_x_0 fields are read-only and the value is considered as all-zero. Thus RBSZ_13_y field indicates the buffer size in terms of locations (with the width same as bus-width).
2-1 RBSZ_x_0	Receive Buffer size Low RBSZ[13:0] splits into two fields RBSZ_13_y and RBSZ_x_0. The RBSZ_x_0 is the lower field whose width is based on data bus width of the configuration. The field width is of 2, 3, or 4 bits for 32-bit, 64-bit, or 128-bit data bus width respectively. This field is read-only (RO).
0 SR	Start or Stop Receive Indicates whether to start or stop receive. When this field is 1, it indicates that the DMA tries to acquire the descriptor from the receive list and processes the incoming packets. DMA tries to acquire descriptor from either of these positions: • The current position in the list. This is the address which DMA_CH0_RxDesc_List_Address sets. • The position at which the receive process was previously stopped. If DMA does not own the current descriptor, the reception is suspended and DMA_CH0_Status[RBU] = 1. The start receive command is effective only when the reception stops. If the command is issued before setting DMA_CH0_RxDesc_List_Address, the DMA behavior is unpredictable.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field becomes 0, it indicates that the receive DMA operation stops after the transfer of the current packet. The next descriptor position in the receive list is saved, and it becomes the current position after the receive process restarts. The stop receive command is effective only when the receive process is in the running (waiting for receive packet) or suspended state. 0b - Stop 1b - Start

72.18.248 DMA Channel 0 Tx Descriptor List Address (DMA_CH0_TxDesc_List_Address)

Offset

Register	Offset
DMA_CH0_TxDesc_List_Address	1114h

Function

Specifies DMA to the start of transmit descriptor list. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). DMA internally converts it to bus width aligned address by making the corresponding LSB low.

You can write to this register only when the transit DMA stops, that is, DMA_CH0_Tx_Control[ST] = 0. When stopped, write this register with a new descriptor list address. When DMA_CH0_Tx_Control[ST] = 1, DMA uses the newly-programmed descriptor base address. If this register does not change when DMA_CH0_Tx_Control[ST] = 0, DMA uses the descriptor address where it was stopped earlier.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TDESLA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TDESLA														Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2	Start of Transmit List

Table continues on the next page...

Table continued from the previous page...

Field	Function
TDESLA	Contains the base address of the first descriptor in the transmit descriptor list. DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

72.18.249 DMA Channel 0 Rx Descriptor List Address (DMA_CH0_RxDesc_List_Address)

Offset

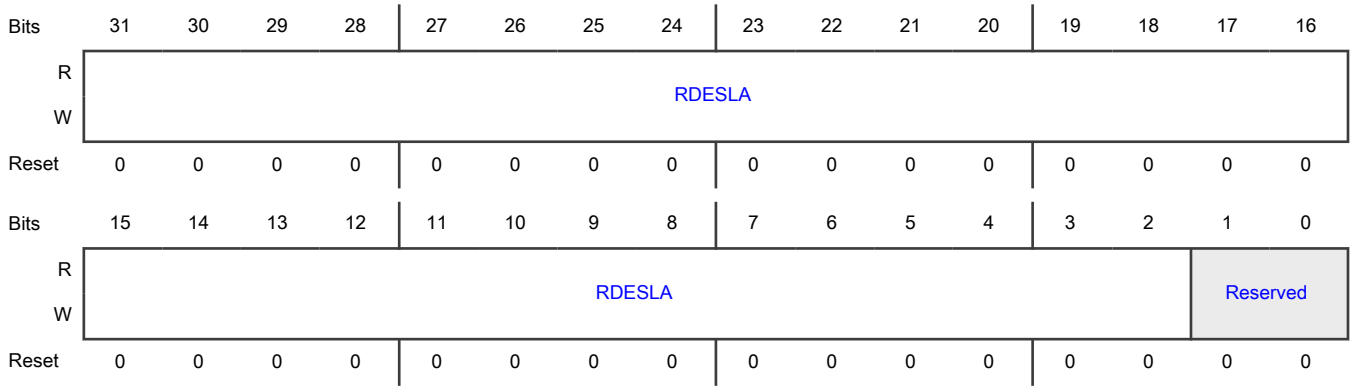
Register	Offset
DMA_CH0_RxDesc_List_Address	111Ch

Function

Specifies DMA to the start of receive descriptor list. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). DMA internally converts it to bus width aligned address by making the corresponding LS bits low. You can write to this register only when reception stops. When stopped, write this register before the receive start command is given. You can write to this register only when Rx DMA stops, that is, DMA_CH0_Rx_Control[SR] = 0. When stopped, write this register with a new descriptor list address.

When DMA_CH0_Rx_Control[SR] = 1, it indicates that DMA takes the newly programmed descriptor base address.

Diagram



Fields

Field	Function
31-2 RDESLA	Start of Receive List Contains the base address of the first descriptor in the receive descriptor list. DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

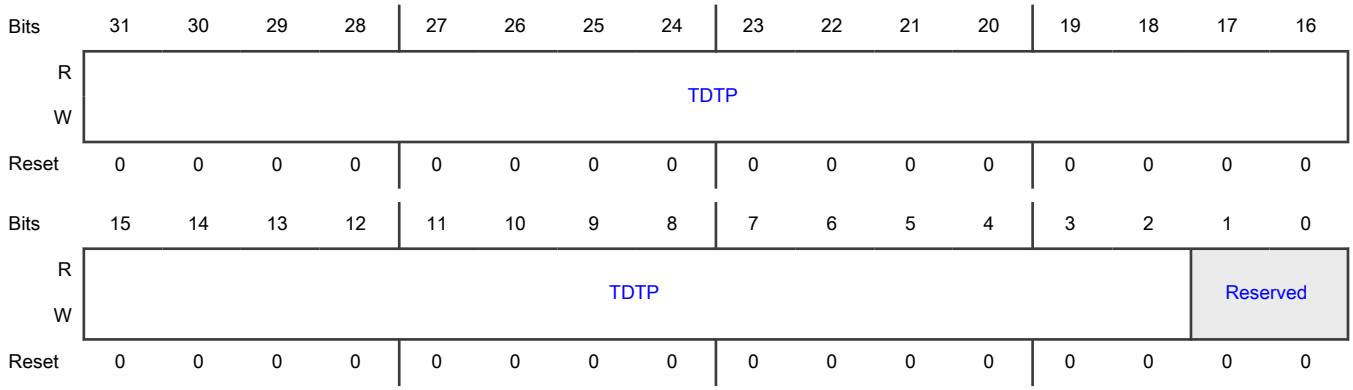
72.18.250 DMA Channel 0 Tx Descriptor Tail Pointer (DMA_CH0_TxDesc_Tail_Pointer)

Offset

Register	Offset
DMA_CH0_TxDesc_Tail_Pointer	1120h

Function
Points to an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-2	Transmit Descriptor Tail Pointer

Table continues on the next page...

Table continued from the previous page...

Field	Function
TDTP	Contains the tail pointer for the transit descriptor ring. You must write the tail pointer to add more descriptors to the transit channel. The hardware tries to transmit all packets to the descriptors referenced between the head and the tail pointer registers. The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

72.18.251 DMA Channeli 0 Rx Descriptor List Pointer (DMA_CH0_RxDesc_Tail_Pointer)

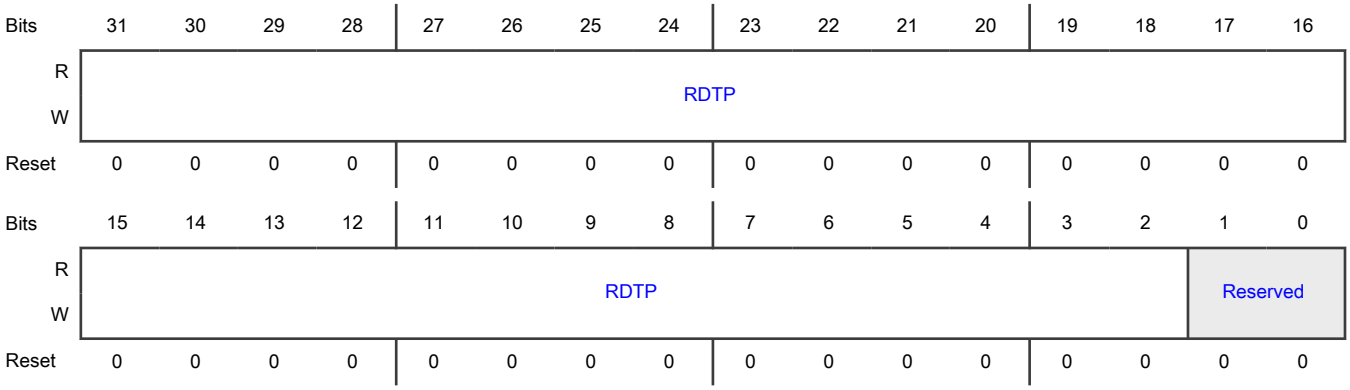
Offset

Register	Offset
DMA_CH0_RxDesc_Tail_Pointer	1128h

Function

Specifies an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-2 RDTP	Receive Descriptor Tail Pointer

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Contains the tail pointer for the receive descriptor ring. You must write the tail pointer to add more descriptors to the receive channel. The hardware tries to write all received packets to the descriptors referenced between the head and the tail pointer registers. The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

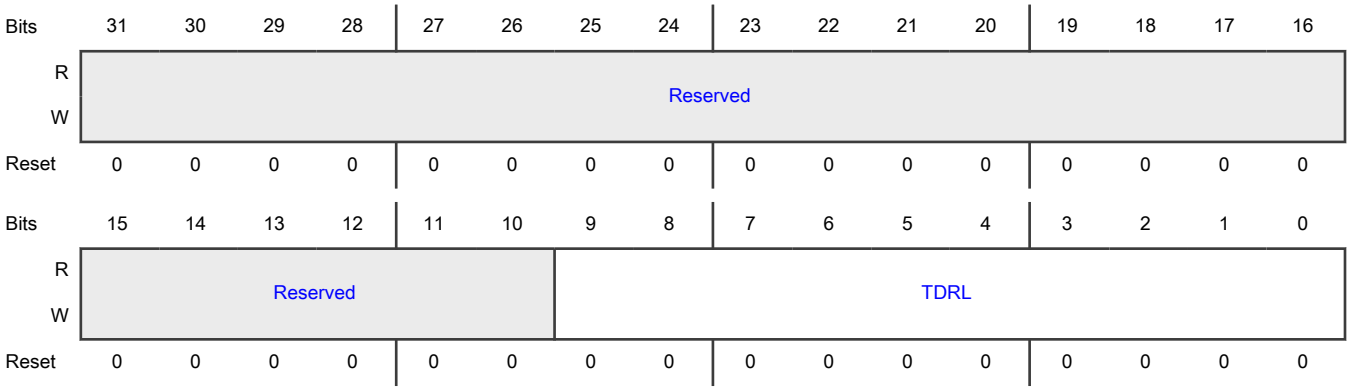
72.18.252 DMA Channel 0 Tx Descriptor Ring Length (DMA_CH0_TxDesc_Ring_Length)

Offset

Register	Offset
DMA_CH0_TxDesc_Ring_Length	112Ch

Function
Contains the length of the transmit descriptor ring.

Diagram



Fields

Field	Function
31-10 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-0 TDRL	Transmit Descriptor Ring Length Sets the maximum number of transmit descriptors in the circular descriptor ring. The maximum number of descriptors are limited to 1K descriptors. NXP recommends a minimum ring descriptor length of 4. For example, you can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. Program this field to a value of 0x9, if you want to have 10 descriptors.

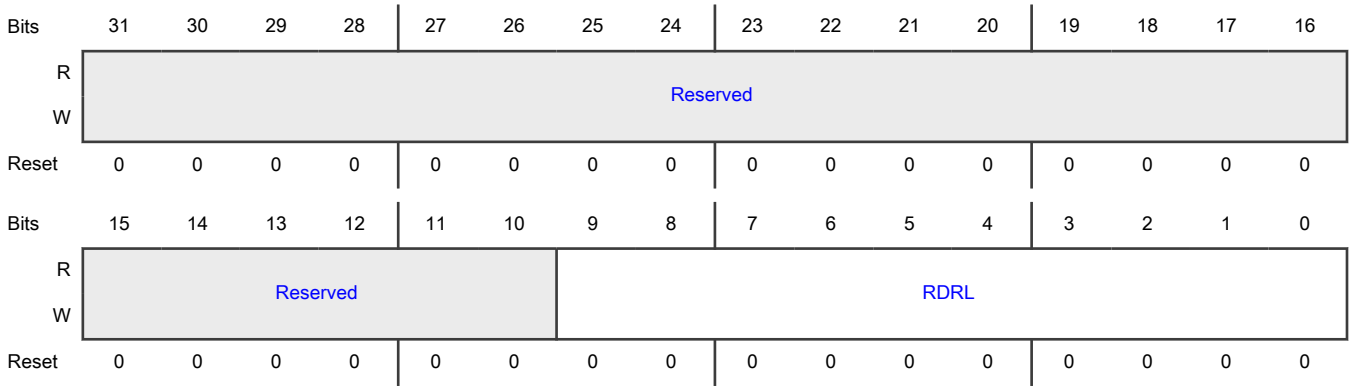
72.18.253 DMA Channel 0 Rx Descriptor Ring Length (DMA_CH0_RxDesc_Ring_Length)

Offset

Register	Offset
DMA_CH0_RxDesc_Ring_Length	1130h

Function
Contains the length of the receive descriptor circular ring.

Diagram



Fields

Field	Function
31-10 —	Reserved.
9-0 RDRL	Receive Descriptor Ring Length Sets the maximum number of receive descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. For example, you can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. Program this field to a value of 0x9, if you want to have 10 descriptors.

72.18.254 DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)

Offset

Register	Offset
DMA_CH0_Interrupt_Enable	1134h

Function
Enables the interrupts which the status register report.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NIE	AIE	CDEE	FBEE	ERIE	ETIE	RWTE	RSE	RBUE	RIE	Reserved			TBUE	TXSE	TIE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved.
15 NIE	Normal Interrupt Summary Enable Enables or disables the normal interrupt summary. When this field is 1, it enables the normal interrupt summary. This field also enables the following interrupts in DMA_CH0_Status: -Bit 0 - Transmit interrupt Bit 2 - Transmit buffer unavailable Bit 6 - Receive interrupt Bit 11 - Early receive interrupt When this field becomes 0, it disables the normal interrupt summary. 0b - Disable 1b - Enable
14	Abnormal Interrupt Summary Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
AIE	Enables or disables the abnormal interrupt summary. When this field is 1, it enables the abnormal interrupt summary. This field also enables the following interrupts in DMA_CH0_Status: Bit 1 - Transmit process stopped Bit 7 - Rx buffer unavailable Bit 8 - Receive process stopped Bit 9 - Receive watchdog timeout Bit 10 - Early transmit interrupt Bit 12 - Fatal bus error Bit 13 - Context descriptor error When this field becomes 0, it disables the abnormal interrupt summary. 0b - Disable 1b - Enable
13 CDEE	Context Descriptor Error Enable Enables or disables the context descriptor error. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the descriptor error interrupt. When this field becomes 0, it disables the descriptor error interrupt. 0b - Disable 1b - Enable
12 FBEE	Fatal Bus Error Enable Enables or disables the fatal bus error. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the fatal bus error interrupt. When this field becomes 0, it disables the fatal bus error interrupt. 0b - Disable 1b - Enable
11 ERIE	Early Receive Interrupt Enable Enables or disables the early receive interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the early receive interrupt. When this field becomes 0, it disables the early receive interrupt. 0b - Disable 1b - Enable
10 ETIE	Early Transmit Interrupt Enable Enables or disables the early transmit interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the early transmit interrupt. When this field becomes 0, it disables the early transmit interrupt. 0b - Disable 1b - Enable
9 RWTE	Receive Watchdog Timeout Enable Enables or disables the receive watchdog timeout. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the receive watchdog timeout interrupt. When this field becomes 0, it disables the receive watchdog timeout interrupt. 0b - Disable 1b - Enable
8 RSE	Receive Stopped Enable Enables or disables the receive stopped interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the receive stopped interrupt. When this field becomes 0, it disables the receive stopped interrupt. 0b - Disable 1b - Enable
7 RBUE	Receive Buffer Unavailable Enable Enables or disables the receive buffer unavailable interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the receive buffer unavailable interrupt. When this field becomes 0, it disables the receive buffer unavailable interrupt. 0b - Disable 1b - Enable
6 RIE	Receive Interrupt Enable Enables or disables the receive interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the receive interrupt. When this field becomes 0, it disables the receive interrupt. 0b - Disable 1b - Enable
5-3 —	Reserved.
2 TBUE	Transmit Buffer Unavailable Enable Enables or disables the transmit buffer unavailable interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the transmit buffer unavailable interrupt. When this field becomes 0, it disables the transmit buffer unavailable interrupt. 0b - Disable 1b - Enable
1 TXSE	Transmit Stopped Enable Enables or disables the transmit stopped interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the transmission stopped interrupt. When this field becomes 0, it disables the transmission stopped interrupt. 0b - Disable 1b - Enable
0 TIE	Transmit Interrupt Enable Enables or disables the transmit interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the transmit interrupt. When this field becomes 0, it disables the transmit interrupt. 0b - Disable 1b - Enable

72.18.255 DMA Channel 0 Rx Interrupt Watchdog Timer (DMA_CH0_Rx_Interrupt_Watchdog_Timer)

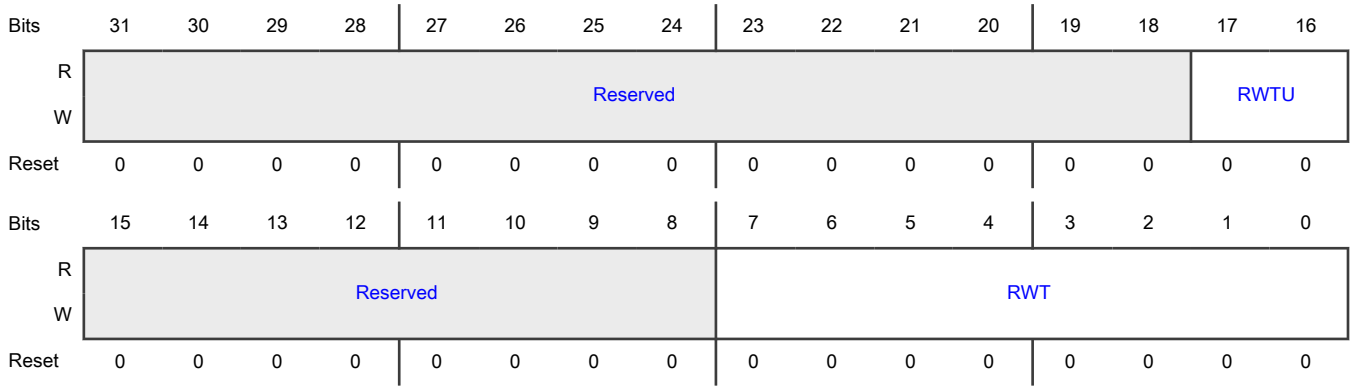
Offset

Register	Offset
DMA_CH0_Rx_Interrupt_Watchdog_Timer	1138h

Function

Indicates the watchdog timeout for receive interrupt (RI) from DMA. When you write this register with a non-zero value, it enables the watchdog timer for the RI bit of the DMA_CHi_Status register.

Diagram



Fields

Field	Function
31-18 —	Reserved.
17-16 RWTU	Receive Interrupt Watchdog Timer Count Units Indicates the system clock cycles number corresponding to one unit in DMA_CH0_Rx_Interrupt_Watchdog_Timer[RWT]. 2'b00 - 256 2'b01 - 512 2'b10 - 1024 2'b11 - 2048 For example, when DMA_CH0_Rx_Interrupt_Watchdog_Timer[RWT] = 2 and DMA_CH0_Rx_Interrupt_Watchdog_Timer[RWTU] = 1, the watchdog timer sets for 2*512=1024 system clock cycles.
15-8 —	Reserved.
7-0 RWT	Receive Interrupt Watchdog Timer Count Indicates the number of system clock cycles, multiplied by factor which DMA_CH0_Rx_Interrupt_Watchdog_Timer[RWTU] indicates, for which you set the watchdog timer. The watchdog timer triggers with the programmed value after the receive DMA completes the packet transfer for which the RI bit is not set in the DMA_CH(#i)_Status register, because of the setting of interrupt enable bit in the corresponding descriptor RDES3[30]. When the watchdog timer runs out, it indicates that the RI bit is 1 and the timer stops. The watchdog timer resets when the RI bit is 1 because of automatic setting of RI per the interrupt enable bit RDES3[30] of any received packet.

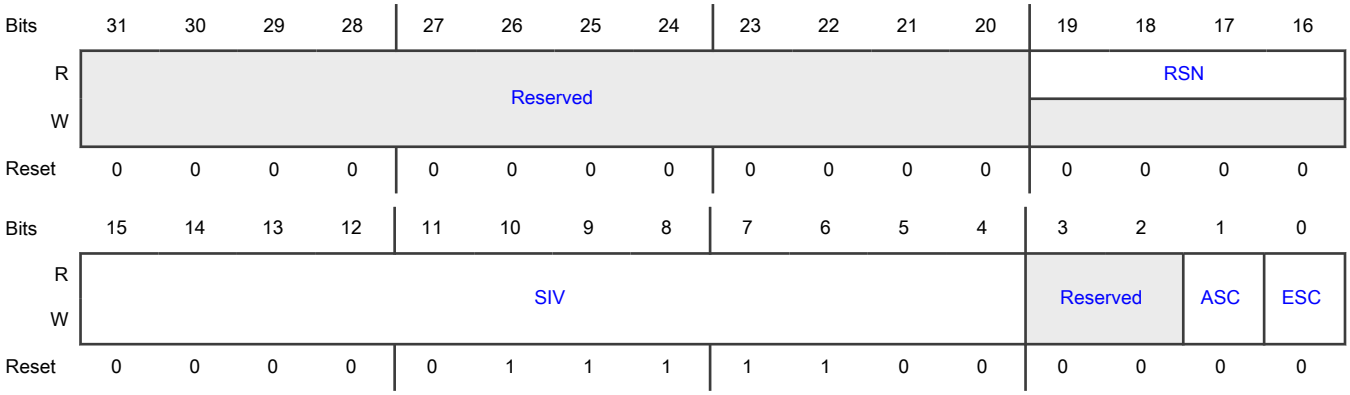
72.18.256 DMA Channel 0 Slot Function Control Status (DMA_CH0_Slot_Function_Control_Status)

Offset

Register	Offset
DMA_CH0_Slot_Function_Control_Status	113Ch

Function
Contains the control bits for slot function and the status for transmit path.

Diagram



Fields

Field	Function
31-20 —	Reserved.
19-16 RSN	Reference Slot Number Provides the current value of the reference slot number in DMA. It is used for slot comparison.
15-4 SIV	Slot Interval Value Controls the period of the slot interval in which the TxDMA fetches the scheduled packets. A value of 0 specifies the slot interval of 1 us while the maximum value 4095 specifies the slot interval of 4096 us. The default or reset value is 0x07C which corresponds to slot interval of 125 us
3-2 —	Reserved.
1 ASC	Advance Slot Check Indicates whether the advance slot check is enabled. When this field is 1, it enables DMA to fetch the data from the buffer when the slot number (SLOTNUM) programmed in the transit descriptor is either:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Equal to the reference slot number given in DMA_CH0_Slot_Function_Control_Status[RSN], or Ahead of the reference slot number by up to two slots. This field is applicable only when DMA_CH0_Slot_Function_Control_Status[ESC] = 1. 0b - Disabled 1b - Enabled
0 ESC	Enable Slot Comparison Indicates whether the slot comparison is enabled. When this field is 1, it enables the checking of the slot numbers programmed in the transit descriptor with the current reference given in DMA_CH0_Slot_Function_Control_Status[RSN]. DMA fetches the data from the corresponding buffer only when the slot number is either: • Equal to the reference slot number, or • Ahead of the reference slot number by one slot. When this field becomes 0, it disables the checking of the slot numbers. DMA fetches the data immediately after you process the descriptor. **NOTE** You must not enable the UFO (UDP Fragmentation over IPv4)/TSO/USO along with TBS/AVB slot number check. The UFO/TSO/USO involves multiple packets/segments/ fragments transmission for single packet received from application and the slot number check are applicable for fetching only first segment/fragment. As a result it might be difficult for you to specify slot number for subsequent packets. 0b - Disabled 1b - Enabled

72.18.257 DMA Channel 0 Current Application Transmit Descriptor (DMA_CH0_Current_App_TxDesc)

Offset

Register	Offset
DMA_CH0_Current_App_TxDesc	1144h

Function
Specifies the current transmit descriptor which DMA reads.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURTDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURTDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURTDESAPTR	Application Transmit Descriptor Address Pointer Indicates that DMA updates this pointer during the transit operation. This pointer clears when reset.

72.18.258 DMA Channel 0 Current Application Receive Descriptor (DMA_CH0_Current_App_RxDesc)

Offset

Register	Offset
DMA_CH0_Current_App_RxDesc	114Ch

Function

Specifies the current receive descriptor which DMA read.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURRDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURRDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURRDESAPTR	Application Receive Descriptor Address Pointer Indicates that the DMA updates this pointer during the receive operation. This pointer clears when reset.

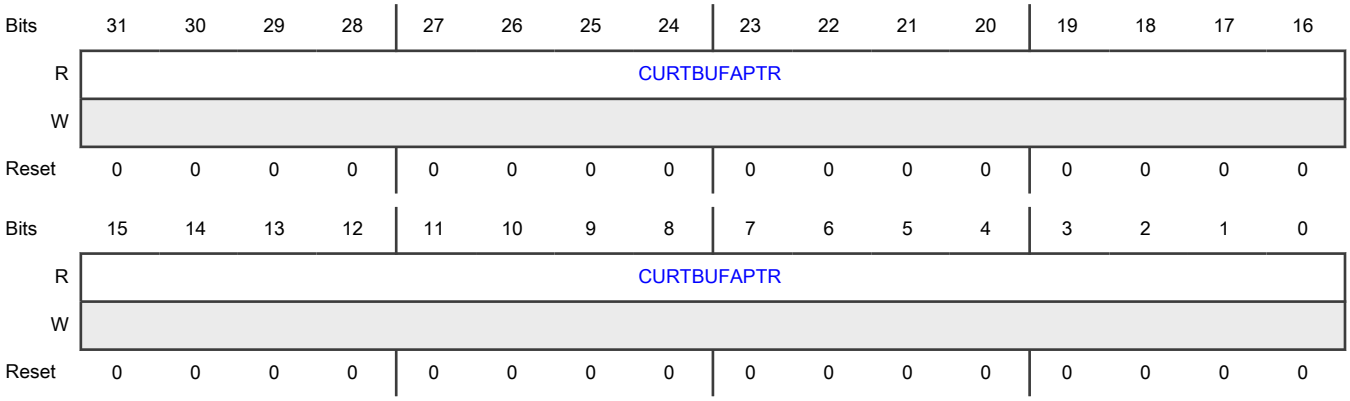
72.18.259 DMA Channel 0 Current Application Transmit Descriptor (DMA_CH0_Current_App_TxBuffer)

Offset

Register	Offset
DMA_CH0_Current_App_TxBuffer	1154h

Function
Specifies the current transit buffer address which DMA reads.

Diagram



Fields

Field	Function
31-0 CURTBUFAPTR	Application Transmit Buffer Address Pointer Indicates that DMA updates this pointer during the transit operation. This pointer clears when reset.

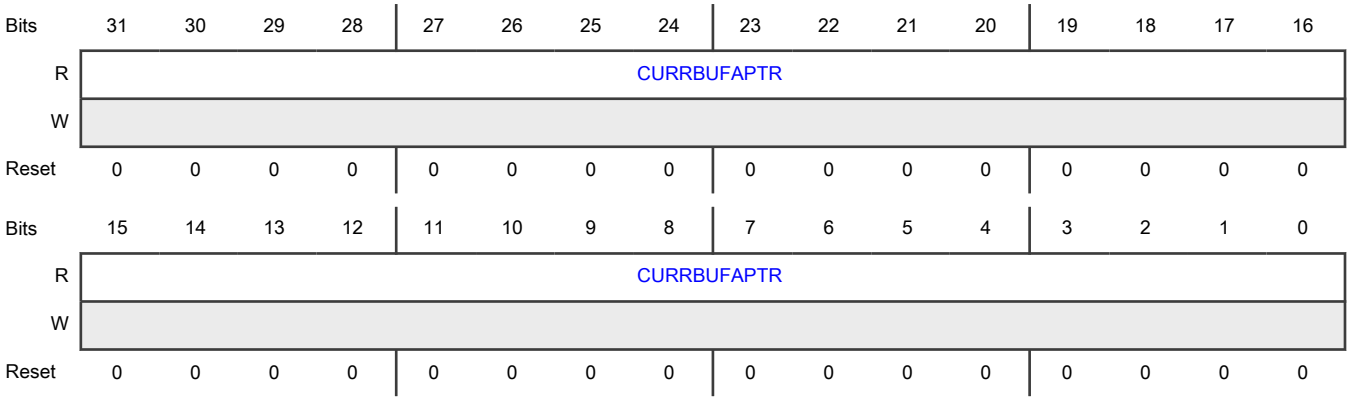
72.18.260 DMA Channel 0 Current Application Receive Buffer (DMA_CH0_Current_App_RxBuffer)

Offset

Register	Offset
DMA_CH0_Current_App_RxBuffer	115Ch

Function
Specifies the current receive buffer address which DMA read.

Diagram



Fields

Field	Function
31-0 CURRBUFAPTR	Application Receive Buffer Address Pointer Indicates that DMA updates this pointer during the receive operation. This pointer clears when reset.

72.18.261 DMA Channel 0 Status (DMA_CH0_Status)

Offset

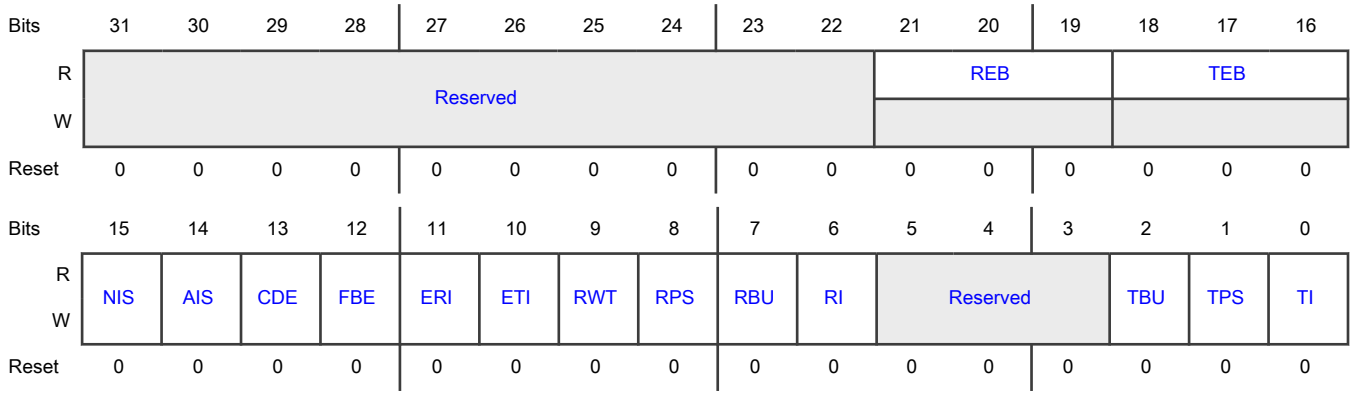
Register	Offset
DMA_CH0_Status	1160h

Function
Indicates that the software driver (application) reads the status register during an interrupt service routine or polling to determine the DMA status.

NOTE

The number of DMA_CH(#i)_Status register in the configuration is the higher of number of receive DMA channels and transit DMA channels.

Diagram



Fields

Field	Function
31-22 —	Reserved.
21-19 REB	Rx DMA Error Bits Indicates the type of error that causes a bus error. For example, error response on the AHB or AXI interface. Bit 21 1'b1 - Error during data transfer by Rx DMA 1'b0 - No Error during data transfer by Rx DMA Bit 20 1'b1 - Error during descriptor access 1'b0 - Error during data buffer access Bit 19 1'b1 - Error during read transfer 1'b0 - Error during write transfer This field is valid only when DMA_CH0_Status[FBE] = 1. This field does not generate an interrupt.
18-16 TEB	Tx DMA Error Bits Indicates the type of error that causes a bus error. For example, error response on the AHB or AXI interface. Bit 18 1'b1 - Error during data transfer by Tx DMA 1'b0 - No Error during data transfer by Tx DMA Bit 17 1'b1 - Error during descriptor access 1'b0 - Error during data buffer access

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Bit 16 1'b1 - Error during read transfer 1'b0 - Error during write transfer This field is valid only when DMA_CH0_Status[FBE] = 1. This field does not generate an interrupt.
15 NIS	Normal Interrupt Summary Indicates whether the normal interrupt summary status is detected. This field is the logical OR of the following bits when you enables the corresponding interrupt bits in DMA_CH0_Interrupt_Enable: Bit 0 - Transmit interrupt Bit 2 - Transmit buffer unavailable Bit 6 - Receive interrupt Bit 11 - Early receive interrupt Only unmasked bits (interrupts for which interrupt enable sets in DMA_CH0_Interrupt_Enable) affect this field. This is a sticky bit. You must clear this field by writing 1 to it each time a corresponding field which sets this field clears. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
14 AIS	Abnormal Interrupt Summary Indicates whether an abnormal interrupt summary status is detected. This field is the logical OR of the following bits when you enables the corresponding interrupt bits in DMA_CH0_Interrupt_Enable: Bit 1 - Transmit process stopped Bit 7 - Receive buffer unavailable Bit 8 - Receive process stopped Bit 10 - Early transmit interrupt Bit 12 - Fatal bus error Bit 13 - Context descriptor error Only unmasked bits affect this field. This is a sticky bit. You must clear this field by writing 1 to it each time a corresponding field, which sets this field, clears. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected
13 CDE	Context Descriptor Error Indicates whether the context descriptor error status is detected. This field indicates that the DMA Tx/Rx engine receives a descriptor error, which indicates an invalid context in the middle of packet flow (intermediate descriptor) or all one's descriptor in transit case and on receive side it indicates DMA has read a descriptor with either of the buffer address as ones which is considered to be invalid. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
12 FBE	Fatal Bus Error Indicates whether the fatal bus error status is detected. This field indicates that a bus error occurred (as described in the EB field). When this field is 1, it indicates that the corresponding DMA channel engine disables all bus accesses. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
11 ERI	Early Receive Interrupt Indicates whether an early receive interrupt status is detected. When this field is 1, it indicates that the RxDMA completes the packet data transfer to the memory. In configs supporting ERIC, this field is 1 only after the receive DMA fills a complete receive buffer with packet data, when DMA_CH0_Rx_Control[ERIC] = 0. This field is 1 after every burst transfer of data from the receive DMA to the buffer, when DMA_CH0_Rx_Control[ERIC] = 1. Writing 1 to RI field automatically clears this field. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
10 ETI	Early Transmit Interrupt Indicates whether an early transmit interrupt status is detected. When this field is 1, it indicates that the TxDMA completes the packet data transfer to the MTL TXFIFO memory.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	In configs supporting ERIC: this field is 1 only after the transit DMA transfers a complete packet to MTL, when DMA_CH0_Tx_Control[ETIC] = 0. This field is 1 after packet data transfers from buffer completes (partial) in the transmit descriptor in which IOC = 1, when DMA_CH0_Tx_Control[ETIC] = 1. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
9 RWT	Receive Watchdog Timeout Indicates whether the receive watchdog timeout status is detected. Asserts when you receives a packet with length greater than 2,048 bytes (10,240 bytes when Jumbo Packet mode is enabled). 0b - Not detected 1b - Detected
8 RPS	Receive Process Stopped Indicates whether the receive process stopped status is detected. Asserts when the receive process enters the stopped state. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
7 RBU	Receive Buffer Unavailable Indicates whether the receive buffer unavailable status is detected. This field indicates that the application owns the next descriptor in the receive list, and DMA cannot acquire it. The receive process is suspended. To resume processing receive descriptors, the application must change the ownership of the descriptor and issue a receive poll demand command. If this command is not issued, the receive process resumes when you receive the next recognized incoming packet. In ring mode, the application must advance the receive descriptor tail pointer register of a channel. This field is 1 only when DMA owns the previous receive descriptor. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
6 RI	Receive Interrupt Indicates that the receive interrupt status is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field indicates that the packet reception is complete. When packet reception completes, bit 31 of RDES3 resets in the last descriptor, and you can update the specific packet status information in the descriptor. The reception remains in the running state. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
5-3 —	Reserved.
2 TBU	Transmit Buffer Unavailable Indicates whether the transmit buffer unavailable status is detected. This field indicates that the application owns the next descriptor in the transmit list, and the DMA cannot acquire it. Transmission is suspended. DMA_Debug_Status0[TPS0] explains the transmit process state transitions. To resume processing the transmit descriptors, the application must perform these steps: 1. Change the ownership of the descriptor by writing 1 to bit 31 of TDES3. 2. Issue a transmit poll demand command. For Ring mode, the application must advance the transmit descriptor tail pointer register of a channel. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
1 TPS	Transmit Process Stopped Indicates whether the transmit process stopped status is detected. This field is 1 when the transmission stops. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
0 TI	Transmit Interrupt Indicates whether the transmit interrupt status is detected. This field indicates that the packet transmission is complete. When transmission completes, Bit 31 of TDES3 resets in the last descriptor, and you can update the specific packet status information in the descriptor.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected

72.18.262 DMA Channel 0 Miss Frame Counter (DMA_CH0_Miss_Frame_Cnt)

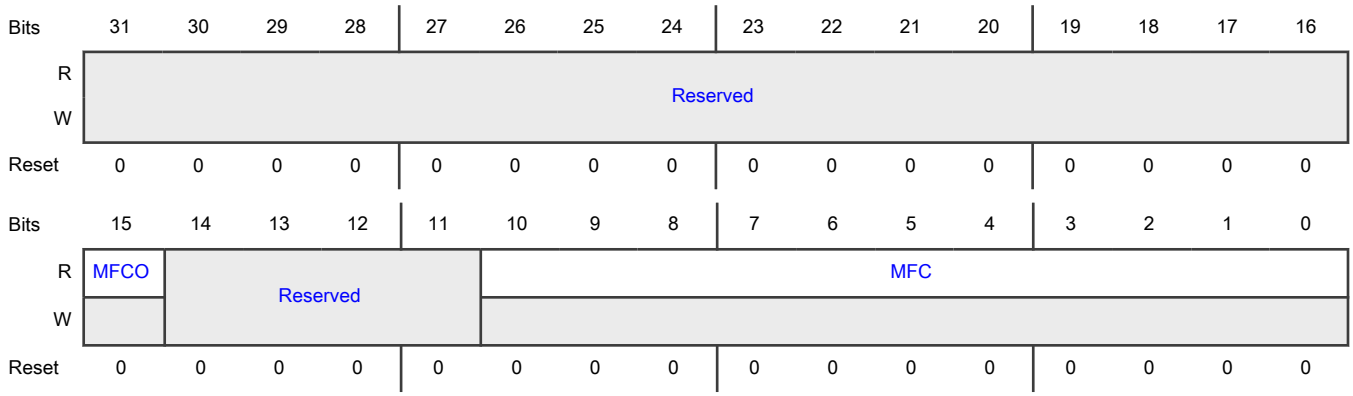
Offset

Register	Offset
DMA_CH0_Miss_Frame_Cnt	1164h

Function

Indicates the number of packet counter that DMA drops either due to bus error or due to programming RPF field in DMA_CH${i}_Rx_Control register.

Diagram



Fields

Field	Function
31-16 —	Reserved.
15 MFCO	Overflow status of the MFC Counter Indicates whether the miss frame counter overflow has occurred. When this field is 1, it indicates that the MFC counter does not increment further. This field is 0 when this register is read.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not occurred 1b - Occurred
14-11 —	Reserved.
10-0 MFC	Dropped Packet Counters Indicates the number of packet counters that DMA drops either because of bus error or because of programing RPF field in DMA_CH${i}_Rx_Control register. This field is 0 when this register is read. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

72.18.263 DMA Channel 0 Rx Parser Accept Count (DMA_CH0_RXP_Accept_Cnt)

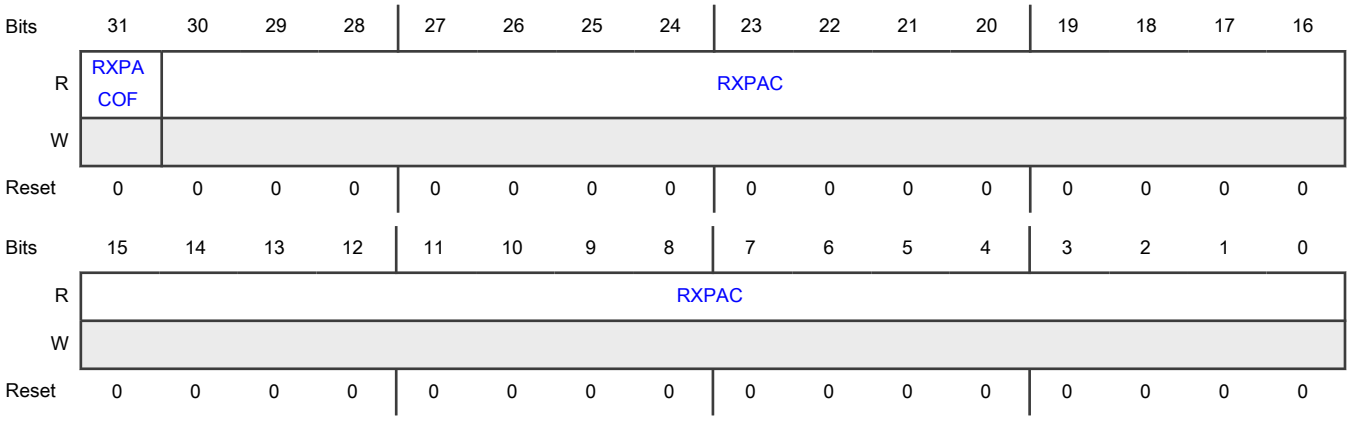
Offset

Register	Offset
DMA_CH0_RXP_Accept_Cnt	1168h

Function

Provides the count of the number of frames which the receive parser accept.

Diagram



Fields

Field	Function
31 RXPACOF	Rx Parser Accept Counter Overflow Bit Indicates whether the receive parser accept counter overflow has occurred. When this field is 1, it indicates that the RXPAC counter field has crossed the maximum limit. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not occurred 1b - Occurred
30-0 RXPAC	Rx Parser Accept Counter Implements whenever a receive parser accept a packet because AF = 1. The counter clears when the register is read.

72.18.264 DMA Channel 0 Rx ERI Count (DMA_CH0_RX_ERI_Cnt)

Offset

Register	Offset
DMA_CH0_RX_ERI_Cnt	116Ch

Function
Provides the count of the number of times ERI is asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				ECNT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11-0 ECNT	ERI Counter Indicates that when ERIC bit of DMA_CH(#i)_RX_Control register is 1, this counter increments for burst transfer which the receive DMA completes from the start of packet transfer. This field becomes 0 at the start of new packet.

72.18.265 DMA Channel 1 Control (DMA_CH1_Control)

Offset

Register	Offset
DMA_CH1_Control	1180h

Function

Specifies the MSS value for segmentation, length to skip between two descriptors, and also the features such as header splitting and 8xPBL mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved							Reserved	Reserved				DSL			Reserved	PBLx8
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved.
24 —	Reserved.
23-21	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
20-18 DSL	Descriptor Skip Length Specifies the Word, Dword, or Lword number (depending on the 32-bit, 64-bit, or 128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of the current descriptor to the start of the next descriptor. When this field is 0, it indicates that DMA takes the descriptor table as contiguous.
17 —	Reserved.
16 PBLx8	8xPBL mode Indicates whether the 8xPBL mode is enabled. When this field is 1, it indicates that you must program the PBL value in Bits[21:16] in DMA_CH(#i)_Tx_Control and Bits[21:16] in DMA_CH(#i)_Rx_Control and multiply it eight times. Therefore, DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. 0b - Disabled 1b - Enabled
15-14 —	Reserved.
13-0 —	Reserved.

72.18.266 DMA Channel 1 Tx Control (DMA_CH1_Tx_Control)

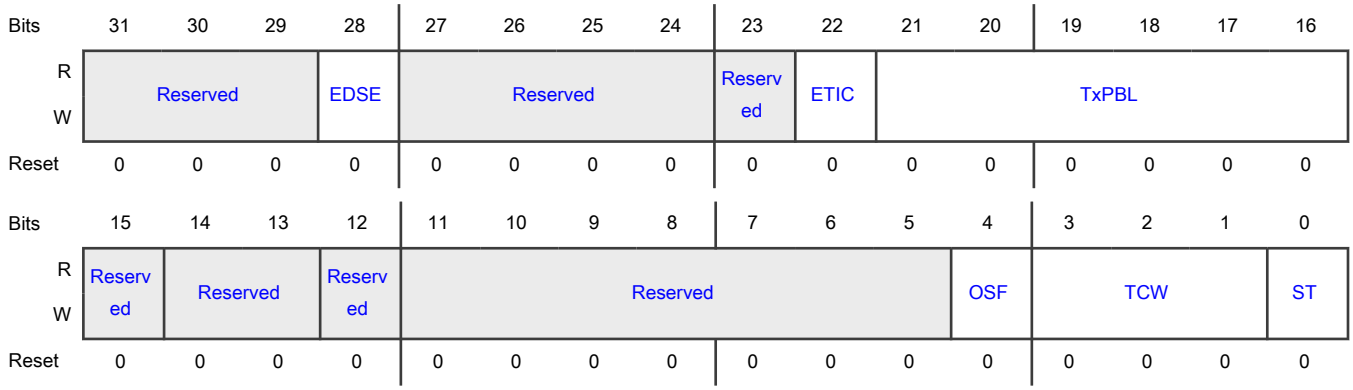
Offset

Register	Offset
DMA_CH1_Tx_Control	1184h

Function

Controls the transmit features such as PBL, TCP segmentation, and transmit channel weights.

Diagram



Fields

Field	Function
31-29 —	Reserved.
28 EDSE	Enhanced Descriptor Enable Indicates whether an enhanced descriptor is enabled. When this field is 1, it indicates that the corresponding channel uses enhanced descriptors that are 32 Bytes for both normal and context descriptors. When this field becomes 0, it indicates that the corresponding channel uses the descriptors that are 16 Bytes. 0b - Disabled 1b - Enabled
27-24 —	Reserved.
23 —	Reserved.
22 ETIC	Early Transmit Interrupt Control Indicates whether an early transmit interrupt is enabled. When this field is 1, it indicates that an early transmit interrupt (ETI) status sets after the data transfer completes from buffers of a transmit descriptor in which IOC bit (TDES2[31]) sets. When this field becomes 0, it indicates that the ETI sets only after a complete packet transfers to the MTL TX FIFO memory. 0b - Disabled 1b - Enabled
21-16	Transmit Programmable Burst Length

Table continues on the next page...

Table continued from the previous page...

Field	Function
TxPBL	Indicates the maximum number of beats to transfer in one DMA block data. DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of these values: 1, 2, 4, 8, 16, or 32. Any other value results in an undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Write 1 to DMA_CH0_Control[PBLx8]. 2. Write 1 to this field. **NOTE** The maximum value of this field must be less than or equal to half the transit queue size (TQS field of MTL_TxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the transit queue has space to store at least another Tx PBL worth of data while the MTL Tx Queue controller transfers data to MAC. For example, in 64-bit data width configurations the total locations in transit queue of size 512 bytes is 64. You must program TxPBL and 8xPBL to less than or equal to 32.
15 —	Reserved.
14-13 —	Reserved.
12 —	Reserved.
11-5 —	Reserved.
4 OSF	Operate on Second Packet Indicates whether an operation on second packet is enabled. When this field is 1, it instructs DMA to process the second packet of the transmit data even before the first packet status is obtained. 0b - Disabled 1b - Enabled
3-1 TCW	Transmit Channel Weight Indicates the weight assigned to the corresponding transmit channel. When reset completes, this field becomes 0 for all channels by default, resulting in equal weights to all channels.
0 ST	Start or Stop Transmission Command Indicates whether to start or stop transmission command. When this field is 1, it indicates that the transmission is placed in the running state. DMA checks the transmit list at the current position to transmit a packet.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DMA tries to acquire descriptor from either of the following positions: • The current position in the list. This is the base address of the transmit list which DMA_CH0_TxDesc_List_Address sets. • The position at which the transmission was previously stopped If DMA does not own the current descriptor, the transmission enters the suspended state and DMA_CH0_Status[TBU] = 1. The start transmission command is effective only when the transmission stops. If you issue the command before writing 1 to DMA_CH0_TxDesc_List_Address, then DMA behavior is unpredictable. When this field becomes 0, it indicates that the transmission process is placed in the stopped state after the transmission of the current packet completes. The next descriptor position in the transmit list is saved, and it becomes the current position when the transmission restarts. To change the list address, you must program DMA_CH0_TxDesc_List_Address with a new value when this field becomes 0. The new value is considered when this field is 1 again. The stop transmission command is effective only when the transmission of the current packet completes or the transmission is in the suspended state. 0b - Stop 1b - Start

72.18.267 DMA Channel 1 Rx Control (DMA_CH1_Rx_Control)

Offset

Register	Offset
DMA_CH1_Rx_Control	1188h

Function

Controls the receive features such as PBL, buffer size, and extended status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RPF	Reserved			Reserved				Reserved	ERIC	RxPBL					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	RBSZ_13_y												RBSZ_x_0		SR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 RPF	Rx Packet Flush Indicates whether the receive packet flush is enabled. When this field is 1, it indicates that the module automatically flushes the packet from the receive queues destined to this DMA receive channel, when it stops. When this field remains 1 and the software driver re-starts DMA, the packets residing in the receive queues that were received when this RxDMA was stopped, flushes out. Route the packets to the RxDMA that the MAC receives after the RxDMA re-starts. The flushing takes place on the read side of the receive queue. When this field is 0, it indicates that the module do not flush the packet in the receive queue destined to this RxDMA channel when it is in stop state. This might in turn cause head-of-line blocking in the corresponding RxQueue. **NOTE** The stopping of packet flow from a Rx DMA Channel to the application by writing 1 to this field works only when there is one-to-one mapping of receive queue to the receive DMA channels. In Dynamic mapping mode, writing 1 to this field in any DMA_CH(#i)_Rx_Control register might flush packets from unintended receive queues which are destined to the stopped Rx DMA Channel. 0b - Disabled 1b - Enabled
30-28 —	Reserved.
27-24 —	Reserved.
23 —	Reserved.
22 ERIC	Early Receive Interrupt Control Indicates whether an early receive interrupt control status is enabled. When this field is 1, it indicates that an early receive interrupt (ERI) status sets after every burst transfer of data from the receive DMA to the buffer completes. When this field becomes 0, it indicates that ERI sets only after the RxDMA fills the complete buffer is filled. 0b - Disabled 1b - Enabled
21-16 RxPBL	Receive Programmable Burst Length Indicates the maximum number of beats to transfer in one DMA block data. DMA always attempts max burst as specified in PBL, each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in an undefined behavior. To transfer more than 32 beats, perform the following steps:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1. Write 1 to DMA_CH0_Control[PBLx8]. 2. Write 1 to RxPBL. **NOTE** The maximum value of RxPBL must be less than or equal to half the receive queue size (RQS field of MTL_RxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the receive queue has space to store at least another Rx PBL worth of data when the receive DMA transfers a block of data. For example, in 64-bit data width configurations the total locations in receive queue of size 512 bytes is 64, so you must program RxPBL and 8xPBL to less than or equal to 32.
15 —	Reserved.
14-3 RBSZ_13_y	Receive Buffer size High Indicates that RBSZ[13:0] splits into two fields higher RBSZ_13_y and lower RBSZ_x_0. RBSZ[13:0] field indicates the size of the receive buffers specified in bytes. The maximum buffer size is limited to 16K bytes. The buffer size is applicable to payload buffers when you enable split headers. **NOTE** The buffer size must be a multiple of 4, 8, or 16 depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively). This is required even if the value of buffer address pointer is not aligned to data bus width. Hence the lower RBSZ_x_0 bits are read-only and the value is considered as all-zero. Thus RBSZ_13_y indicates the buffer size in terms of locations (with the width same as bus-width).
2-1 RBSZ_x_0	Receive Buffer size Low Indicates that RBSZ[13:0] splits into two fields RBSZ_13_y and RBSZ_x_0. RBSZ_x_0 is the lower field whose width is based on data bus width of the configuration. This field width is of 2, 3, or 4 bits for 32-bit, 64-bit, or 128-bit data bus width respectively. This field is read-only (RO).
0 SR	Start or Stop Receive Indicates whether to start or stop receive. When this field is 1, it indicates that DMA tries to acquire the descriptor from the receive list and processes the incoming packets. DMA tries to acquire descriptor from either of the following positions: • The current position in the list. This is the address that DMA_CH0_RxDesc_List_Address sets. • The position at which the receive process was previously stopped If DMA does not own the current descriptor, the reception suspends and DMA_CH0_Status[RBU] = 1. The start receive command is effective only when the reception stops. If the command is issued before writing 1 to DMA_CH0_RxDesc_List_Address, DMA behavior is unpredictable.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field becomes 0, it indicates that the receive DMA operation stops after the transfer of the current packet. The next descriptor position in the receive list is saved, and it becomes the current position after the receive process restarts. The stop receive command is effective only when the receive process is in the running (waiting for Rx packet) or suspended state. 0b - Stop 1b - Start

72.18.268 DMA Channel 1 Tx Descriptor List Address (DMA_CH1_TxDesc_List_Address)

Offset

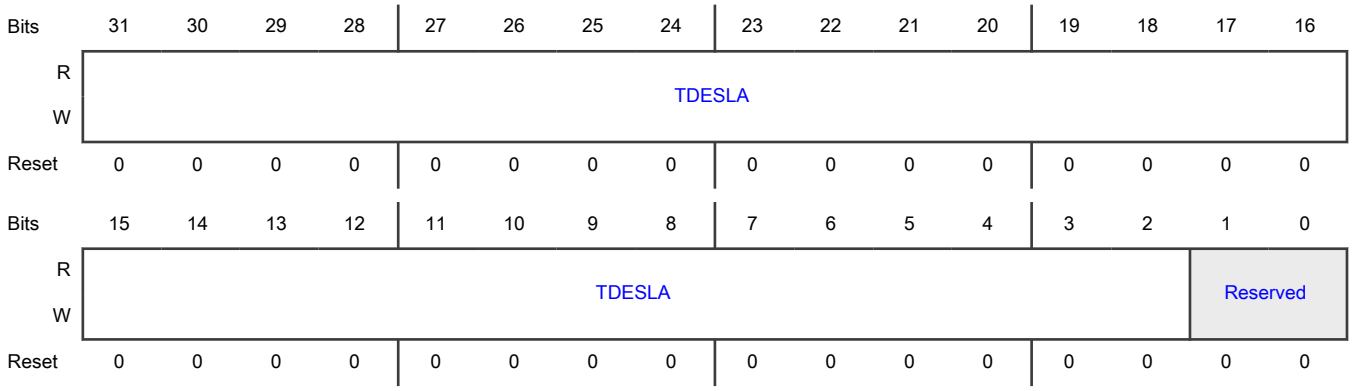
Register	Offset
DMA_CH1_TxDesc_List_Address	1194h

Function

Specifies DMA to the start of transmit descriptor list. The descriptor lists reside in the application's physical memory space and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA converts the corresponding LSB bits to low to internally convert the descriptor lists to bus width aligned address.

You can write to this register only when the transmit DMA stops, that is, DMA_CH0_Tx_Control[ST] = 0. When stopped, you can write this register with a new descriptor list address. When DMA_CH0_Tx_Control[ST] = 1, the DMA takes the newly-programmed descriptor base address. If this register do not change when DMA_CH0_Tx_Control[ST] = 0, DMA takes the descriptor address where it was stopped earlier.

Diagram



Fields

Field	Function
31-2	Start of Transmit List

Table continues on the next page...

Table continued from the previous page...

Field	Function
TDESLA	Contains the first descriptor base address in the transmit descriptor list. DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally assumes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

72.18.269 DMA Channel 1 Rx Descriptor List Address (DMA_CH1_RxDesc_List_Address)

Offset

Register	Offset
DMA_CH1_RxDesc_List_Address	119Ch

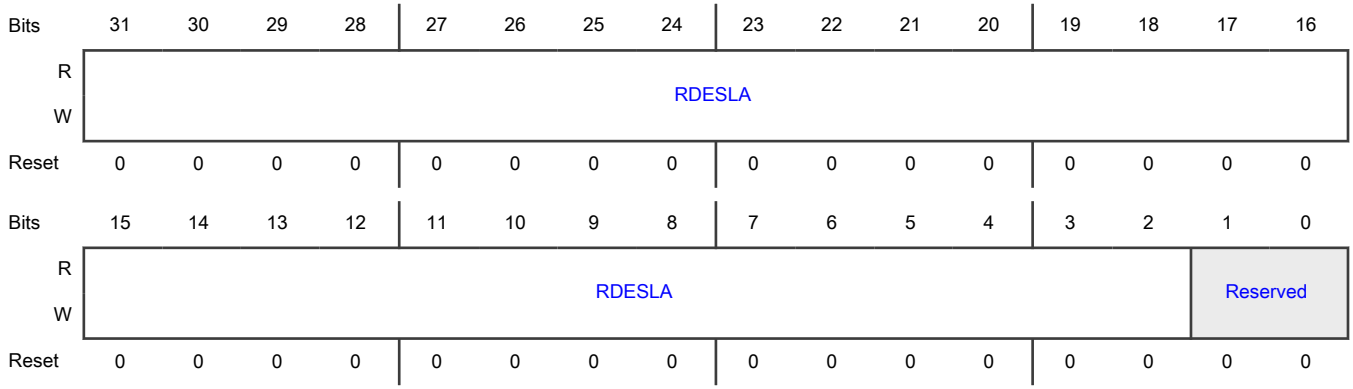
Function

Specifies DMA to the start of receive descriptor list. The descriptor lists resides in the application's physical memory space and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). DMA converts the correspondong LS bits low to internally convert the descriptor lists to bus width aligned address. You can write to this register only when reception stops. When stopped, you must write this register before the receive start command is given.

You can write to this register when receive DMA has stopped, that is, DMA_CH0_Rx_Control[SR] = 0. When stopped, you can this register with a new descriptor list address.

When DMA_CH0_Rx_Control[SR] = 1, DMA takes the newly programmed descriptor base address.

Diagram



Fields

Field	Function
31-2 RDESLA	Start of Receive List Contains the first descriptior base address in the receive descriptor list. DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally assume these bits as all-zero. Therefore, these LSB bits are read-only (RO). The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

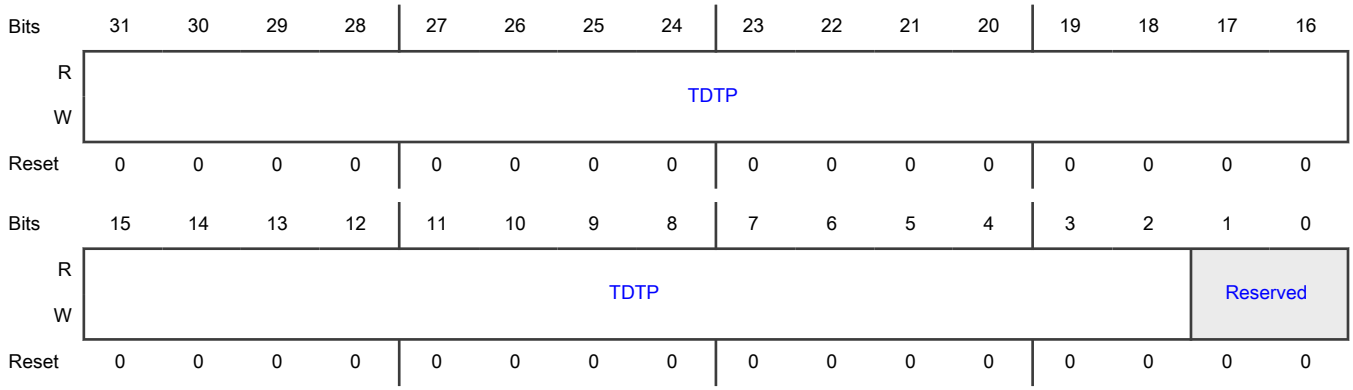
72.18.270 DMA Channel 1 Tx Descriptor Tail Pointer (DMA_CH1_TxDesc_Tail_Pointer)

Offset

Register	Offset
DMA_CH1_TxDesc_Tail_Pointer	11A0h

Function
Specifies to an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-2	Transmit Descriptor Tail Pointer

Table continues on the next page...

Table continued from the previous page...

Field	Function
TDTP	Contains the tail pointer for the transit descriptor ring. The software writes the tail pointer to add more descriptors to the transit channel. The hardware tries to transmit all te packets which the descriptors referenced between the head and the tail pointer registers. The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

72.18.271 DMA Channel 1 Rx Descriptor Tail Pointer (DMA_CH1_RxDesc_Tail_Pointer)

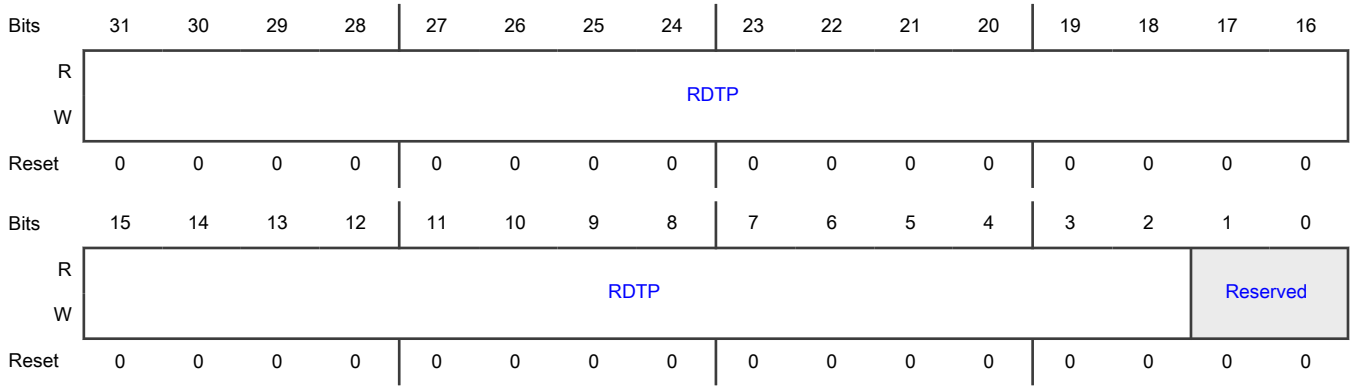
Offset

Register	Offset
DMA_CH1_RxDesc_Tail_Pointer	11A8h

Function

Specifies to an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-2 RDTP	Receive Descriptor Tail Pointer

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Contains the tail pointer for the receive descriptor ring. The software writes the tail pointer to add more descriptors to the receive channel. The hardware tries to write all received packets to the descriptors referenced between the head and the tail pointer registers. The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

72.18.272 DMA Channel 1 Tx Descriptor Ring Length (DMA_CH1_TxDesc_Ring_Length)

Offset

Register	Offset
DMA_CH1_TxDesc_Ring_Length	11ACh

Function
Contains the length of the transmit descriptor ring.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						TDRL									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-0 TDRL	Transmit Descriptor Ring Length Sets the maximum number of transmit descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1 K descriptors. NXP recommends a minimum ring descriptor length of 4. For example, you can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. Program it to a value of 0x9 if you want to have 10 descriptors.

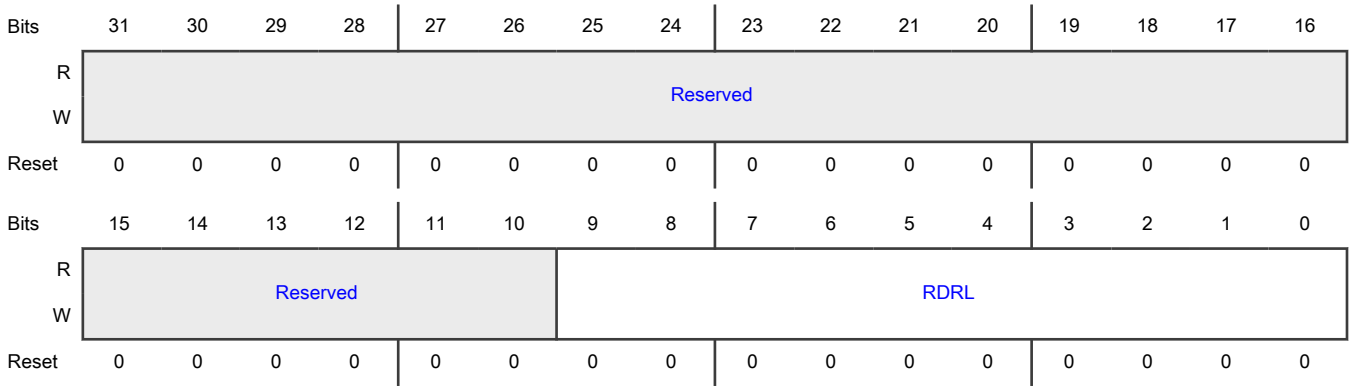
72.18.273 DMA Channel 1 Rx Descriptor Ring Length (DMA_CH1_RxDesc_Ring_Length)

Offset

Register	Offset
DMA_CH1_RxDesc_Ring_Length	11B0h

Function
Contains the length of the receive descriptor circular ring.

Diagram



Fields

Field	Function
31-10 —	Reserved.
9-0 RDRL	Receive Descriptor Ring Length Sets the maximum number of receive descriptors in the circular descriptor ring. The maximum number of descriptors are limited to 1 K descriptors. For example, you can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. Program it to a value of 0x9 if you want to have 10 descriptors.

72.18.274 DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)

Offset

Register	Offset
DMA_CH1_Interrupt_Enable	11B4h

Function
Enables the interrupts which the status register reports.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NIE	AIE	CDEE	FBEE	ERIE	ETIE	RWTE	RSE	RBUE	RIE	Reserved			TBUE	TXSE	TIE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved.
15 NIE	Normal Interrupt Summary Enable Enables or disables the normal interrupt summary. When this field is 1, it enables the normal interrupt summary. This field also enables the following interrupts in DMA_CH0_Status: Bit 0 - Transmit interrupt Bit 2 - Transmit buffer unavailable Bit 6 - Receive interrupt Bit 11 - Early receive interrupt When this field becomes 0, it disables the normal interrupt summary. 0b - Disable 1b - Enable
14	Abnormal Interrupt Summary Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
AIE	Enables or disables the abnormal interrupt summary. When this field is 1, it enables the abnormal interrupt summary. This field also enables the following interrupts in DMA_CH0_Status: Bit 1 - Transmit process stopped Bit 7 - Rx buffer unavailable Bit 8 - Receive process stopped Bit 9 - Receive watchdog timeout Bit 10 - Early transmit interrupt Bit 12 - Fatal bus error Bit 13 - Context descriptor error When this field becomes 0, it disables the abnormal interrupt summary. 0b - Disable 1b - Enable
13 CDEE	Context Descriptor Error Enable Enables or disables the context descriptor error. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the descriptor error interrupt. When this field becomes 0, it disables the descriptor error interrupt. 0b - Disable 1b - Enable
12 FBEE	Fatal Bus Error Enable Enables or disables fatal bus error. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the fatal bus error interrupt. When this field becomes 0, it disables the fatal bus error interrupt. 0b - Disable 1b - Enable
11 ERIE	Early Receive Interrupt Enable Enables or disables the early receive interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the early receive interrupt. When this field becomes 0, it disables the early receive interrupt. 0b - Disable 1b - Enable
10 ETIE	Early Transmit Interrupt Enable Enables or disables the early transmit interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the early transmit interrupt. When this field becomes 0, it disables the early transmit interrupt. 0b - Disable 1b - Enable
9 RWTE	Receive Watchdog Timeout Enable Enables or disables the receive watchdog timeout interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the receive watchdog timeout interrupt. When this field becomes 0, it disables the receive watchdog timeout interrupt. 0b - Disable 1b - Enable
8 RSE	Receive Stopped Enable Enables or disables the receive stopped interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], It enables the receive stopped interrupt. When this field becomes 0, it disables the receive stopped interrupt. 0b - Disable 1b - Enable
7 RBUE	Receive Buffer Unavailable Enable Enables or disables the receive buffer unavailable interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the receive buffer unavailable interrupt. When this field becomes 0, it disables the receive buffer unavailable interrupt. 0b - Disable 1b - Enable
6 RIE	Receive Interrupt Enable Enables or disables the receive interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the receive interrupt. When this field becomes 0, it disables the receive interrupt. 0b - Disable 1b - Enable
5-3 —	Reserved.
2 TBUE	Transmit Buffer Unavailable Enable Enables or disables the transmit buffer unavailable interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the transmit buffer unavailable interrupt. When this field becomes 0, it disables the transmit buffer unavailable interrupt. 0b - Disable 1b - Enable
1 TXSE	Transmit Stopped Enable Enables or disables the transmit stopped interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the transmission stopped interrupt. When this field becomes 0, it disables the transmission Stopped interrupt. 0b - Disable 1b - Enable
0 TIE	Transmit Interrupt Enable Enables or disables the transmit interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the transmit interrupt. When this field becomes 0, it disables the transmit interrupt. 0b - Disable 1b - Enable

72.18.275 DMA Channel 1 Rx Interrupt Watchdog Timer (DMA_CH1_Rx_Interrupt_Watchdog_Timer)

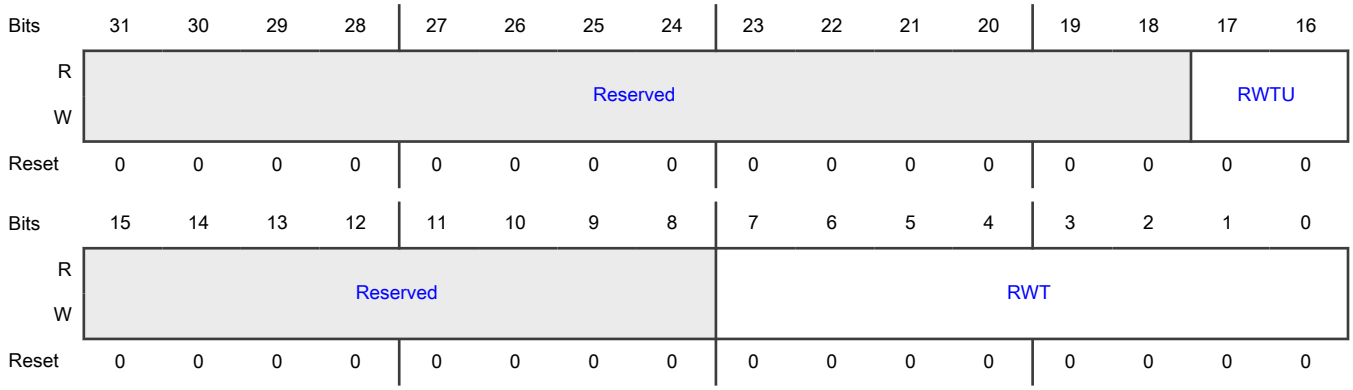
Offset

Register	Offset
DMA_CH1_Rx_Interrupt_Watchdog_Timer	11B8h

Function

Indicates the watchdog timeout for receive interrupt (RI) from DMA. When you write this register with a non-zero value, it enables the watchdog timer for the RI bit of the DMA_CHi_Status register.

Diagram



Fields

Field	Function
31-18 —	Reserved.
17-16 RWTU	Receive Interrupt Watchdog Timer Count Units Indicates the number of system clock cycles corresponding to one unit in DMA_CH1_Rx_Interrupt_Watchdog_Timer[RWT]. 2'b00 - 256 2'b01 - 512 2'b10 - 1024 2'b11 - 2048 For example, the watchdog timer sets for 2*512=1024 system clock cycles, when DMA_CH1_Rx_Interrupt_Watchdog_Timer[RWT] = 2 and DMA_CH1_Rx_Interrupt_Watchdog_Timer[RWTU] = 1.
15-8 —	Reserved.
7-0 RWT	Receive Interrupt Watchdog Timer Count Indicates the number of system clock cycles, multiplied by factor indicated in DMA_CH1_Rx_Interrupt_Watchdog_Timer[RWTU], for which the watchdog timer sets. The watchdog timer triggers with the programmed value after the receive DMA completes the packet transfer for which the RI bit is not 1 in DMA_CH(#i)_Status register, because of the setting of interrupt enable bit in the corresponding descriptor RDES3[30]. RI bit is 1 and the timer stops when the watchdog timer runs out. The watchdog timer becomes 0 when RI bit is 1 because RI sets automatically per the interrupt enable bit RDES3[30] of any received packet.

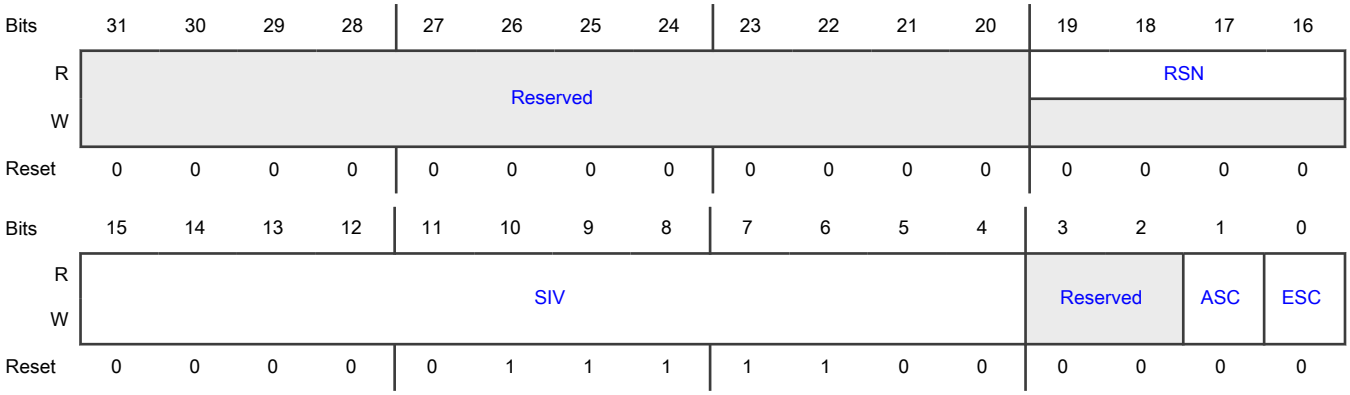
72.18.276 DMA Channel 1 Slot Function Control Status (DMA_CH1_Slot_Function_Control_Status)

Offset

Register	Offset
DMA_CH1_Slot_Function_Control_Status	11BCh

Function
Contains the control field for slot function and the status for transmit path.

Diagram



Fields

Field	Function
31-20 —	Reserved.
19-16 RSN	Reference Slot Number Provides the current value of the reference slot number in DMA. It is used for slot comparison.
15-4 SIV	Slot Interval Value Controls the period of the slot interval in which the TxDMA fetches the scheduled packets. A value of 0 specifies the slot interval of 1 us while the maximum value 4095 specifies the slot interval of 4096 us. The default or reset value is 0x07C which corresponds to slot interval of 125 us.
3-2 —	Reserved.
1 ASC	Advance Slot Check Indicates whether an advance slot check is enabled. When this field is 1, it enables DMA to fetch the data from the buffer when the slot number (SLOTNUM) programmed in the transmit descriptor is:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Equal to the reference slot number given in DMA_CH1_Slot_Function_Control_Status[RSN], or • Ahead of the reference slot number by up to two slots. This field is applicable only when DMA_CH1_Slot_Function_Control_Status[ESC] = 1. 0b - Disabled 1b - Enabled
0 ESC	Enable Slot Comparison Enables slot comparison. When this field is 1, it enables the checking of the slot numbers programmed in the transmit descriptor with the current reference given in DMA_CH1_Slot_Function_Control_Status[RSN]. DMA fetches the data from the corresponding buffer only when the slot number is: • Equal to the reference slot number, or • Ahead of the reference slot number by one slot. When this field becomes 0, it disables the checking of the slot numbers. DMA fetches the data immediately after you process the descriptor. **NOTE** You must not enable the UFO (UDP Fragmentation over IPv4)/TSO/USO along with TBS/AVB slot number check. The UFO/TSO/USO involves multiple packets or segments or fragments transmission for single packet received from application and the slot number check are applicable for fetching only first segment/fragment. As a result it might be difficult for you to specify slot number for subsequent packets. 0b - Disable 1b - Enable

72.18.277 DMA Channel 1 Current Application Transmit Descriptor (DMA_CH1_Current_App_TxDesc)

Offset

Register	Offset
DMA_CH1_Current_App_TxDesc	11C4h

Function
Specifies the current transmit descriptor which DMA reads.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURTDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURTDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURTDESAPTR	Application Transmit Descriptor Address Pointer Indicates that DMA updates this pointer during the transmit operation. This pointer is 0 on reset.

72.18.278 DMA Channel 1 Current Application Receive Descriptor (DMA_CH1_Current_App_RxDesc)

Offset

Register	Offset
DMA_CH1_Current_App_RxDesc	11CCh

Function

Specifies the current receive descriptor which DMA reads.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURRDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURRDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURRDESAPTR	Application Receive Descriptor Address Pointer Indicates that DMA updates this pointer during the receive operation. This pointer is 0 on reset.

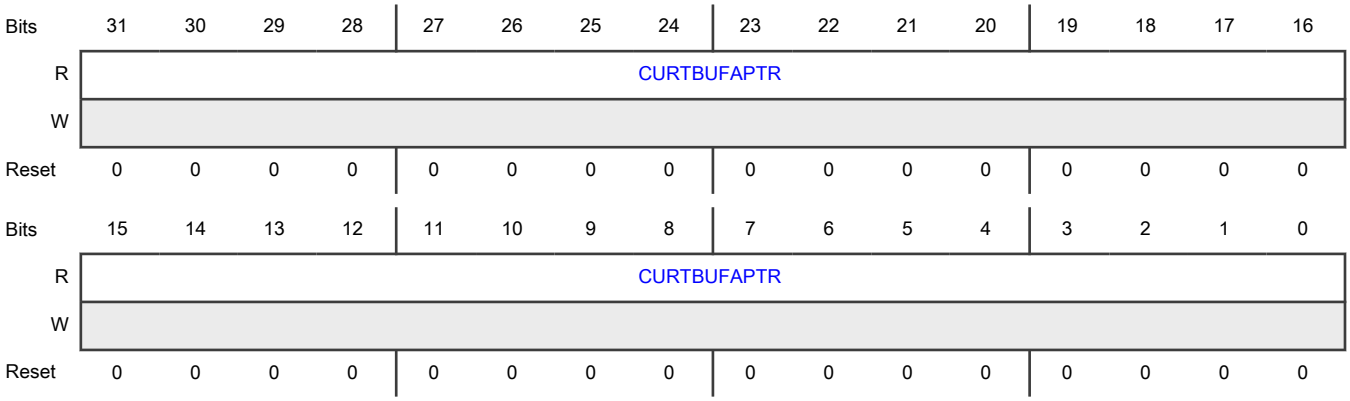
72.18.279 DMA Channel 1 Current Application Transmit Buffer (DMA_CH1_Current_App_TxBuffer)

Offset

Register	Offset
DMA_CH1_Current_App_TxBuffer	11D4h

Function
Specifies the current transmit buffer address which DMA reads.

Diagram



Fields

Field	Function
31-0 CURTBUFAPTR	Application Transmit Buffer Address Pointer Indicates that DMA updates this pointer during the transmit operation. This pointer is 0 on reset.

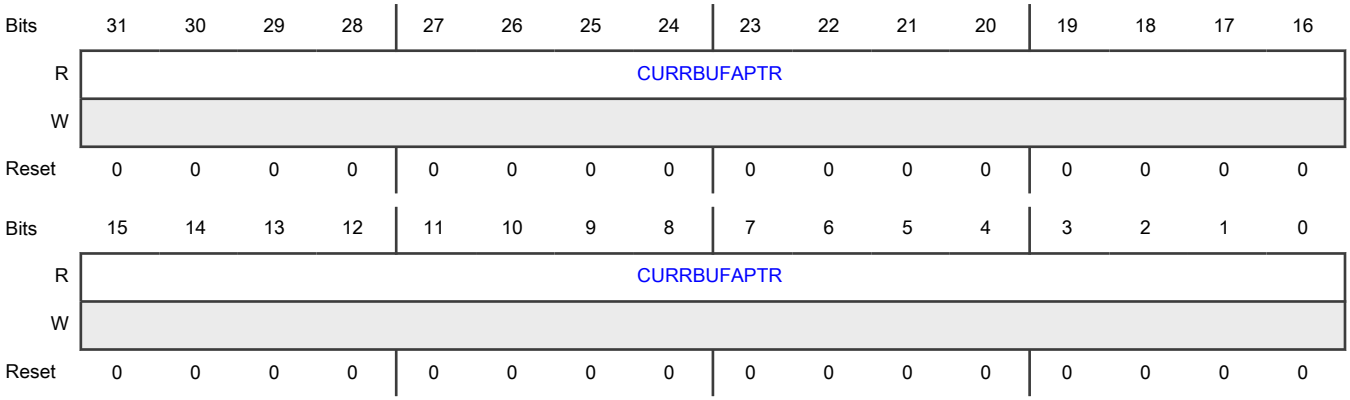
72.18.280 DMA Channel 1 Current Application Receive Buffer (DMA_CH1_Current_App_RxBuffer)

Offset

Register	Offset
DMA_CH1_Current_App_RxBuffer	11DCh

Function
Indicates that DMA_CH0_Current_App_RxBuffer points to the current receive buffer address which the DMA reads.

Diagram



Fields

Field	Function
31-0 CURRBUFAPTR	Application Receive Buffer Address Pointer Indicates that DMA update this pointer during the receive operation. This pointer is 0 on reset.

72.18.281 DMA Channel 1 Status (DMA_CH1_Status)

Offset

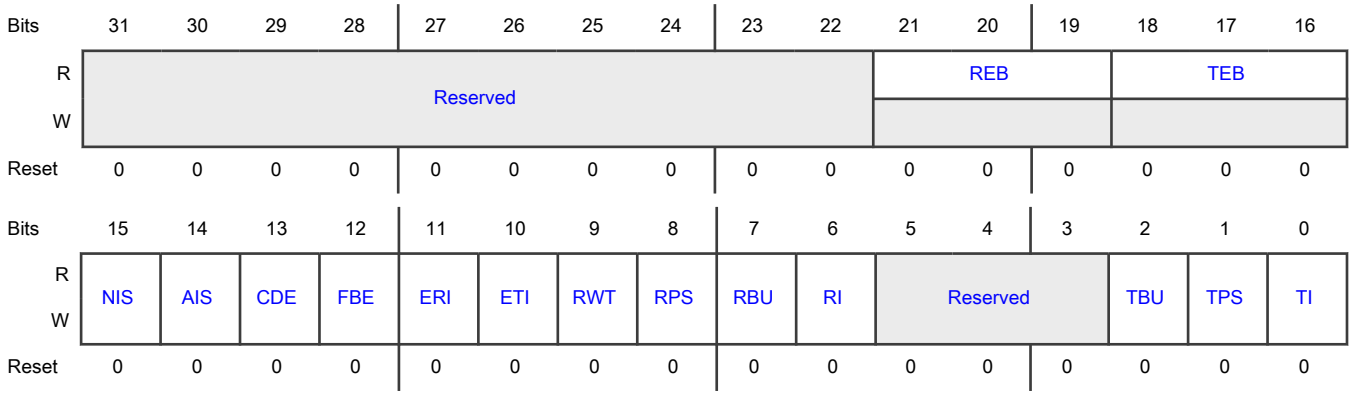
Register	Offset
DMA_CH1_Status	11E0h

Function
Indicates that to determine the status of DMA, the application reads the status register during an interrupt service routine or polling.

NOTE

The number of DMA_CH(#i)_Status register in the configuration is the higher of number of receive DMA Channels and transmit DMA Channels.

Diagram



Fields

Field	Function
31-22 —	Reserved.
21-19 REB	Rx DMA Error Bits Indicates the type of error that causes a bus error. For example, an error response on the AHB or AXI interface. Bit 21 1'b1 - Error during data transfer by Rx DMA 1'b0 - No Error during data transfer by Rx DMA Bit 20 1'b1 - Error during descriptor access 1'b0 - Error during data buffer access Bit 19 1'b1 - Error during read transfer 1'b0 - Error during write transfer This field is valid only when DMA_CH1_Status[FBE] = 1. It does not generate an interrupt.
18-16 TEB	Tx DMA Error Bits Indicates the type of error that causes a bus error. For example, error response on the AHB or AXI interface. Bit 18 1'b1 - Error during data transfer by Tx DMA 1'b0 - No Error during data transfer by Tx DMA Bit 17 1'b1 - Error during descriptor access

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1'b0 - Error during data buffer access Bit 16 1'b1 - Error during read transfer 1'b0 - Error during write transfer This field is valid only when DMA_CH1_Status[FBE] = 1. It does not generate an interrupt.
15 NIS	Normal Interrupt Summary Indicates whether the normal interrupt summary status is detected. The field value is the logical OR of the following bits when the corresponding interrupt bits enable in DMA_CH0_Interrupt_Enable: Bit 0 - Transmit interrupt Bit 2 - Transmit buffer unavailable Bit 6 - Receive interrupt Bit 11 - Early receive interrupt Only unmasked bits (interrupts for which interrupt enable sets in DMA_CH0_Interrupt_Enable) affect this field. This is a sticky field. You must clear this field (by writing 1 to this field) each time a corresponding bit which sets it, clears. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
14 AIS	Abnormal Interrupt Summary Indicates whether an abnormal interrupt summary status is detected. The field value is the logical OR of the following bits when the corresponding interrupt bits enable in DMA_CH0_Interrupt_Enable: Bit 1 - Transmit process stopped Bit 7 - Receive buffer unavailable Bit 8 - Receive process stopped Bit 10 - Early transmit interrupt Bit 12 - Fatal bus error Bit 13 - Context descriptor error Only unmasked bits affect this field. This is a sticky bit. You must clear this field (by writing 1 to this bit) each time a corresponding bit, which sets it, clears.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
13 CDE	Context Descriptor Error Indicates whether the context descriptor error status is detected. This field indicates that the DMA Tx/Rx engine receive a descriptor error, which indicates an invalid context in the middle of packet flow (intermediate descriptor) or all one's descriptor in transmit case and on receive side. It indicates that DMA has read a descriptor with either of the buffer address as ones which you can consider invalid. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
12 FBE	Fatal Bus Error Indicates whether the fatal bus error status is detected. This field indicates that a bus error has occurred (as described in the EB field). When this field is 1, it indicates that the corresponding DMA channel engine disables all bus accesses. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
11 ERI	Early Receive Interrupt Indicates whether an early receive interrupt status is detected. When this field is 1, it indicates that the RxDMA has completed the packet data transfer to the memory. In configs supporting ERIC, When DMA_CH1_Rx_Control[ERIC] = 0, this field is 1 only after the Rx DMA fills a complete receive buffer with packet data. When DMA_CH1_Rx_Control[ERIC] = 1, this field is 1 after every burst data transfer from the receive DMA to the buffer. If DMA_CH1_Status[RI] = 1, this field clears automatically. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
10 ETI	Early Transmit Interrupt Indicates whether an early transmit interrupt status is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1, it indicates that the TxDMA has completed the packet data transfer to the MTL TXFIFO memory. In configs supporting ERIC: When DMA_CH1_Tx_Control[ETIC] = 0, this fied is 1 only after the Tx DMA transfers a complete packet to MTL. When DMA_CH1_Tx_Control[ETIC] = 1, this field is 1 after the packet data transfer completes (partial) from buffers in the Transmit descriptor in which IOC=1. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
9 RWT	Receive Watchdog Timeout Indicates whether a receive watchdog timeout status is detected. This field asserts when it receives a packet with length greater than 2,048 bytes (10,240 bytes when Jumbo Packet mode is enabled). 0b - Not detected 1b - Detected
8 RPS	Receive Process Stopped Indicates whether a receive process stopped status is detected. This field asserts when the receive process enters the stopped state. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
7 RBU	Receive Buffer Unavailable Indicates whether a receive buffer unavailable status is detected. This field indicates that the application owns the next descriptor in the receive list, and the DMA cannot acquire it. The receive process is suspended. To resume processing receive descriptors, the application must change the ownership of the descriptor and issue a receive poll demand command. If this command is not issued, the receive process resumes when you receives the next recognized incoming packet. In Ring mode, the application must advance the receive descriptor tail pointer register of a channel. This field is 1 only when the DMA owns the previous receive descriptor. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
6 RI	Receive Interrupt Indicates whether the receive interrupt status is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field indicates that the packet reception is complete. Bit 31 of RDES3 resets in the last descriptor, and updates the specific packet status information in the descriptor, when the packet reception is complete. The reception remains in the running state. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
5-3 —	Reserved.
2 TBU	Transmit Buffer Unavailable Indicates whether the transmit buffer unavailable status is detected. This field indicates that the application owns the next descriptor in the transmit list, and the DMA cannot acquire it. Transmission is suspended. DMA_Debug_Status0[TPS0] explains the transmit process state transitions. The application must perform these actions, to resume the processing of transmit descriptors: 1. Write 1 to bit 31 of TDES3 to change the ownership of the descriptor. 2. Issue a transmit poll demand command. For Ring mode, the application must advance the transmit descriptor tail pointer register of a channel. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
1 TPS	Transmit Process Stopped Indicates that the transmit process stopped status is detected. This field is 1 when the transmission stops. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
0 TI	Transmit Interrupt Indicates whether the transmit interrupt status is detected. This field indicates that the packet transmission is complete. When transmission completes, bit 31 of TDES3 resets in the last descriptor, and the specific packet status information is updated in the descriptor. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected

72.18.282 DMA Channel 1 Miss Frame Counter (DMA_CH1_Miss_Frame_Cnt)

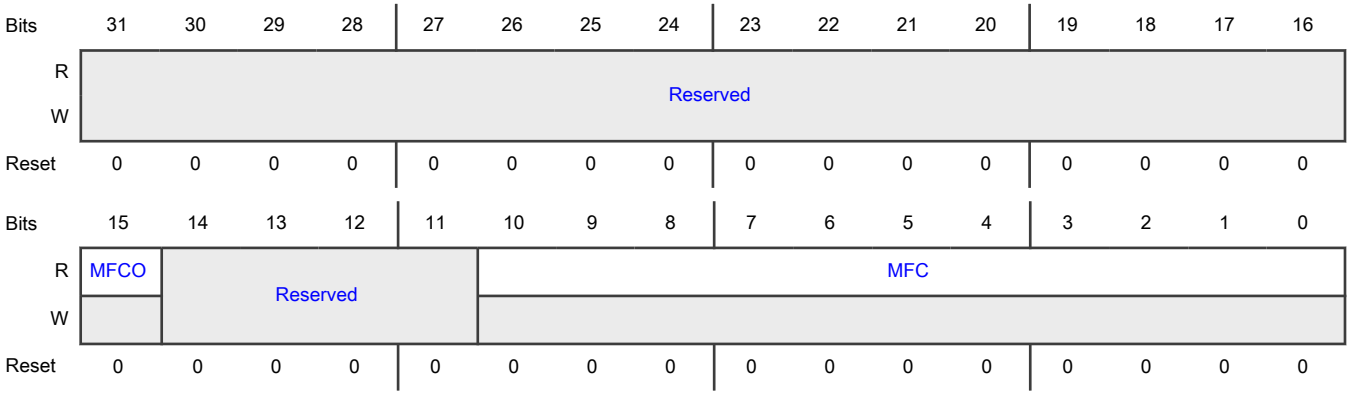
Offset

Register	Offset
DMA_CH1_Miss_Frame_Cnt	11E4h

Function

Provides the number of packet counter that the DMA drops either because of bus error or programming RPF field in DMA_CH${i}_Rx_Control register.

Diagram



Fields

Field	Function
31-16 —	Reserved.
15 MFCO	Overflow status of the MFC Counter Indicates whether the miss frame counter overflow has occurred. When this field is 1, it indicates that the MFC counter does not increments further. This field becomes 0 when this register is read. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not occurred 1b - Occurred
14-11 —	Reserved.
10-0 MFC	Dropped Packet Counters Indicates the number of packet counters that DMA drops either because of bus error or programing RPF field in DMA_CH${i}_Rx_Control register. This counter is 0 when this register is read. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

72.18.283 DMA Channel 1 Rx Parser Accept Count (DMA_CH1_RXP_Accept_Cnt)

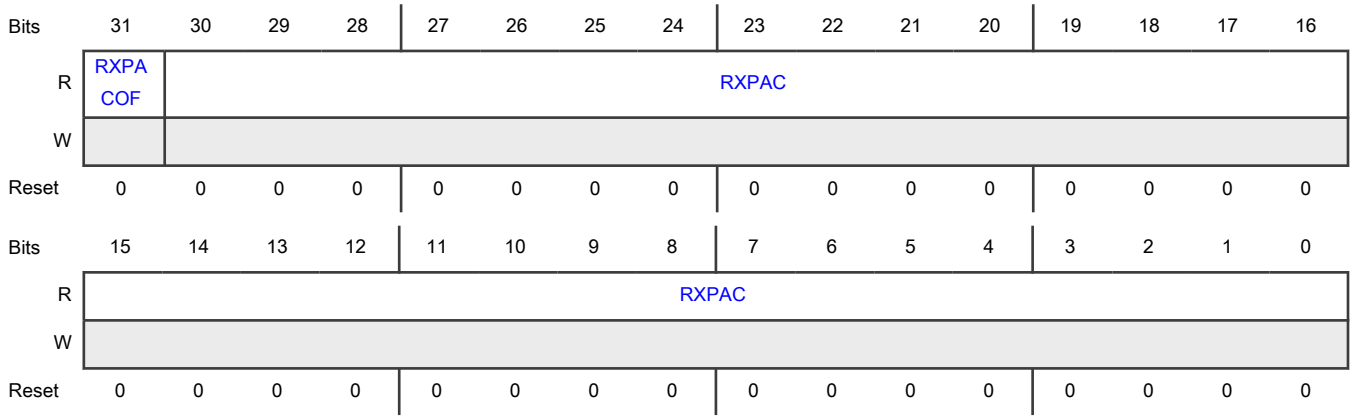
Offset

Register	Offset
DMA_CH1_RXP_Accept_Cnt	11E8h

Function

Provides the count of the number of frames which the receive parser accepts.

Diagram



Fields

Field	Function
31	Rx Parser Accept Counter Overflow Bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXPACOF	Indicates whether the receive parser accept counter overflow have occurred. When this field is 1, it indicates that the DMA_CH1_RXP_Accept_Cnt[RXPAC] has crossed the maximum limit. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not occurred 1b - Occurred
30-0 RXPAC	Rx Parser Accept Counter Implements whenever a receive parser accept a packet because AF = 1. The counter clears when the register is read.

72.18.284 DMA Channel 1 Rx ERI Count (DMA_CH1_RX_ERI_Cnt)

Offset

Register	Offset
DMA_CH1_RX_ERI_Cnt	11ECh

Function

Provides the count of the number of times ERI asserts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				ECNT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11-0 ECNT	ERI Counter ERI Counter When ERIC bit of DMA_CH(#i)_RX_Control register is 1, it indicates that this counter increments for burst transfer which the Rx DMA completes from the start of packet transfer. This counter resets at the start of new packet.

72.19 Glossary

AFM Address filtering module

ARI Application receive interface

ATI Application transmit interface

AVB Audio video bridging

AXI Advanced extensible interface

BTR Base time register

CPT Current presentation time

CRC Cyclic redundancy check

DA Destination address

DCB Data center bridging

DUT Device under test

EOP End of packet

FCS A 32-bit cyclic redundancy check used to detect an in-transit corruption of data. Also marks the end of an Ethernet frame.

FIFO First in first out

GCL Gate control list

GMII Gigabit media independent interface

MAC Media access controller

MCI MAC control tnterface

MDC Management data clock

MDIO Management data input/output

MII Media independent interface

MRI MAC receive interface

MTI MAC transmit interface

MTL MAC transmit layer

PBL Programmable burst length

PHY	The physical interface transceiver that implements the OSI physical layer for an ethernet network. The IEEE 802.3 standard defines it.
RGMII	Reduced gigabit media independent interface
RMII	Reduced media independent interface
RMON	Remote network monitoring
SA	Source address
SFD	Start frame delimiter
SGMII	Serial gigabit media independent interface
SMA	Station management agent
TCP	Transmission control protocol
UDP	User datagram protocol
VLAN	Virtual local area network

Chapter 73
Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)

73.1 Chip-specific LPUART information

73.1.1 LPUART instances and configuration

Table 532. LPUART instances

Instance	MWCT2D17S	MWCT2015S/MWCT2D16S	MWCT2016S
LPUART_0	Yes	Yes	Yes
LPUART_1	Yes	Yes	Yes
LPUART_2	Yes	Yes	Yes
LPUART_3	Yes	Yes	Yes
LPUART_4	Yes	No	Yes
LPUART_5	Yes	No	Yes
LPUART_6	Yes	No	Yes
LPUART_7	Yes	No	Yes
LPUART_8	Yes	No	No
LPUART_9	Yes	No	No
LPUART_10	Yes	No	No
LPUART_11	Yes	No	No
LPUART_12	Yes	No	No
LPUART_13	Yes	No	No
LPUART_14	Yes	No	No
LPUART_15	Yes	No	No

Table 533. LPUART Configuration

Feature	Configuration	
	MWCT2D17S/MWCT2016S	MWCT2014S/MWCT2015S/MWCT2D16S
TX FIFO size	4 Words	16 Words (for instance 0,1) 4 Words (for instance 2,3)
RX FIFO size	4 Words	16 Words (for instance 0,1) 4 Words (for instance 2,3)
Functionality supported	Standard LPUART functionality with MODEM/IrDA	• Standard LPUART functionality with MODEM/IrDA

Table continues on the next page...

Table 533. LPUART Configuration (continued)

Feature	Configuration	
		• MODBUS and PPROFIBUS support[1]
	LIN master and slave operation	

1. Only supported for instance LPUART_0 and LPUART_1.

NOTE

LPUART instances are not available during Standby.

LPUART[0:2]_trg_input is coming from TRGMUX and you should take care of the pulse width of trigger. It should follow the requirement mentioned in section "Peripheral Triggers". These triggers are not present in MWCT2D17S.

73.2 Overview

The Low Power Universal Asynchronous Receiver/Transmitter (LPUART) module provides asynchronous, serial communication capability with external devices. LPUART supports non-return-to-zero (NRZ) encoding format and IrDA compatible infrared (low-speed) SIR format. The LPUART can continue operating while the processor is in low-power mode if an appropriate peripheral clock is available.

73.2.1 Block diagram

The following figure shows the transmitter portion of the LPUART.

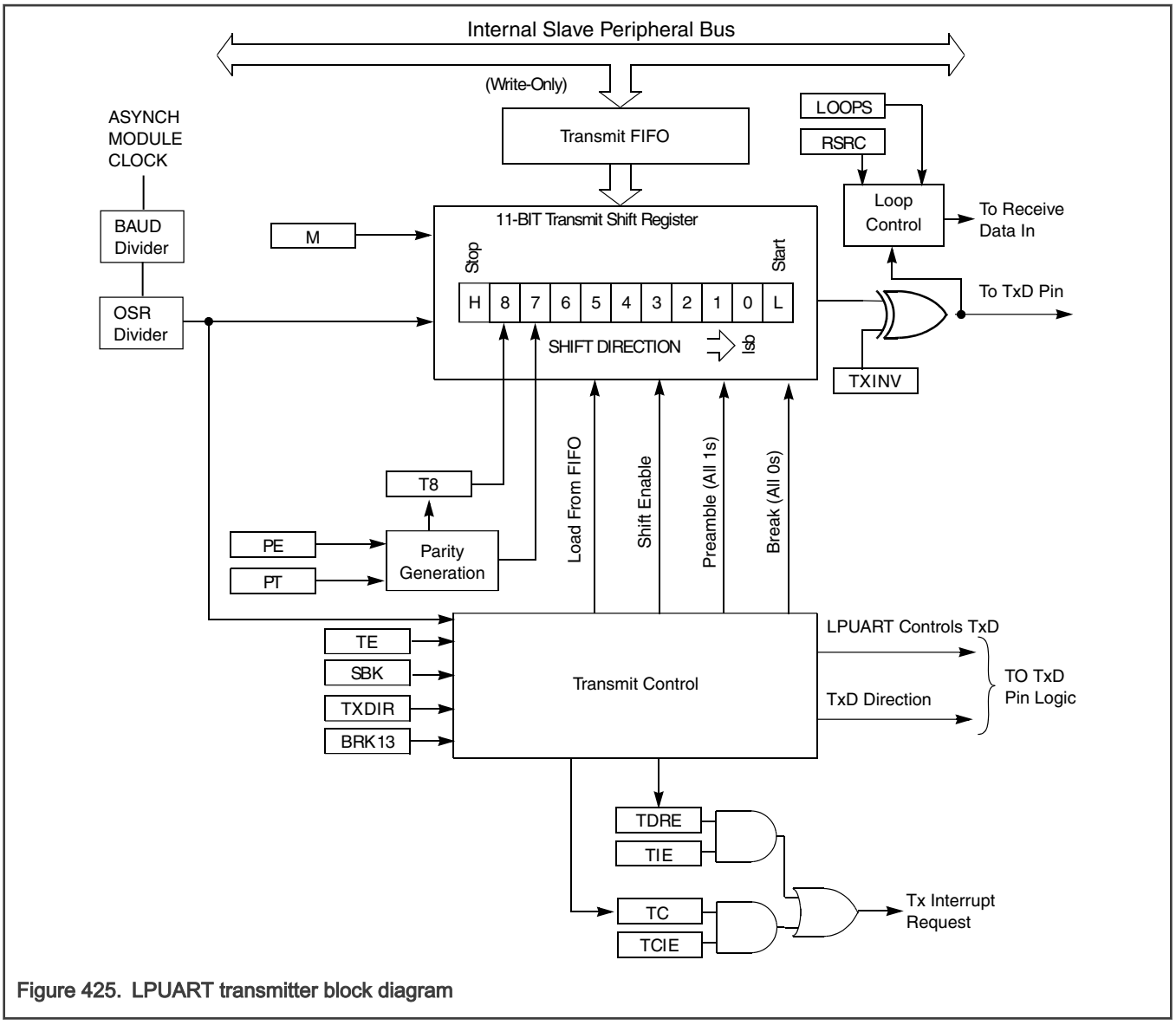


Figure 425. LPUART transmitter block diagram

The following figure shows the receiver portion of the LPUART.

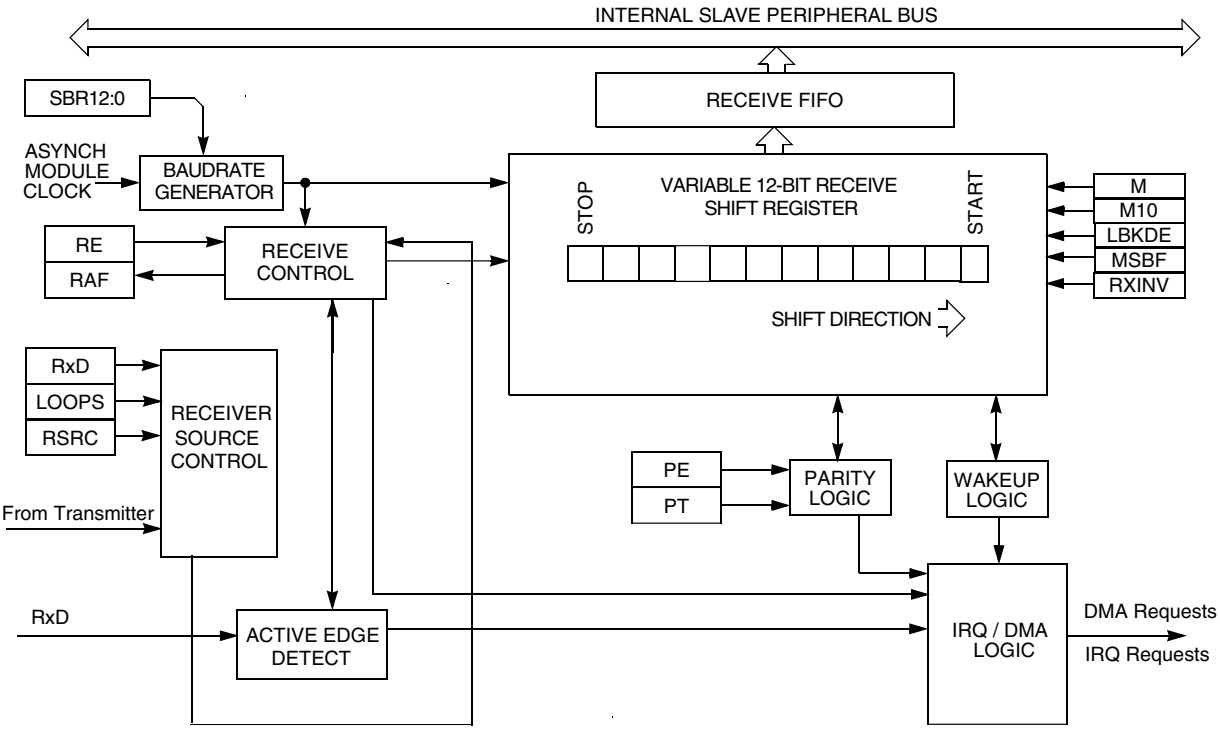


Figure 426. LPUART receiver block diagram

73.2.2 Features

Following are the features of the LPUART module.

- Full-duplex, standard non-return-to-zero (NRZ) format
- Programmable baud rates(13-bit modulo divider) with configurable oversampling ratio from 4x to 32x
- Asynchronous operation of transmit and receive baud rate with respect to the bus clock:
  - Baud rate can be configured independently of the bus clock frequency
  - Supports operation in low-power modes
- Interrupt, DMA or polled operation:
  - Transmit data register empty and transmission complete
  - Receive data register full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
  - Active edge on receive pin
  - Break detect supporting LIN
  - Receive data match
- Hardware parity generation and checking
- Programmable 7-bit, 8-bit, 9-bit or 10-bit character length
- Programmable 1-bit or 2-bit stop bits
- Support for three receiver wakeup methods:
  - Idle line wakeup

  - Address mark wakeup
  - Receive data match
- Automatic address matching to reduce ISR overhead:
  - Address mark matching
  - Idle line address matching
  - Address match start, address match end
- Optional 13-bit/11-bit break character generation
- Configurable idle length detection supporting 1, 2, 4, 8, 16, 32, 64, or 128 idle characters
- Selectable transmitter output and receiver input polarity
- Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with programmable pulse width
- Independent FIFO structure for transmit and receive
  - Separate configurable watermark for receive and transmit requests
  - Option for receiver to assert request after a configurable number of idle characters if receive FIFO is not empty

73.3 Functional description

The LPUART supports full-duplex, asynchronous, NRZ serial communication and comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. The following describes each of the blocks of the LPUART.

73.3.1 Low-power modes

The LPUART remains functional during low power modes, provided the CTRL[DOZEEN] bit is clear and the LPUART functional clock remain enabled. The LPUART can generate an interrupt or DMA request to cause a wakeup from low power modes.

The LPUART can be configured to be disabled in low power modes, when the CTRL[DOZEEN] bit is set. In this case the transmitter and receiver finish transmitting/receiving the current word.

If the LPUART is disabled in low power modes, then it can generate a wakeup via the STAT[RXEDGIF] flag if the receiver detects an active edge.

NOTE

See the chip-specific information to know the specific low-power modes available on your device.

73.3.2 Debug mode

The LPUART remains functional in debug mode.

73.3.3 Baud rate generation

A 13-bit modulus counter in the baud rate generator derives the baud rate for both the receiver and the transmitter. The value from 1 to 8191 written to BAUD[SBR] determines the baud clock divisor for the asynchronous LPUART baud clock. The baud rate clock drives the receiver, while the transmitter is driven by a bit clock which is generated from baud rate clock divided by the over sampling ratio (OSR). Depending on the over sampling ratio, the receiver has an acquisition rate of 4 to 32 samples per bit time.

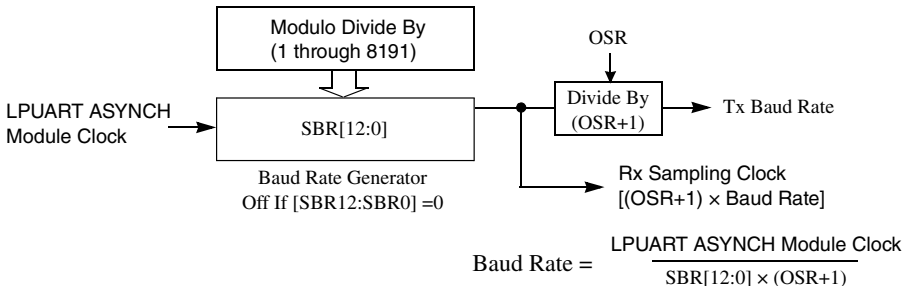


Figure 427. LPUART baud rate generation

Baud rate generation is subject to two sources of error:

- Integer division of the asynchronous LPUART baud clock may not give the exact target frequency.
- Synchronization with the asynchronous LPUART baud clock can cause phase shift.

The baud rate generation is a free-running counter that continues whenever the transmitter or receiver is enabled. The transmitter bit clock continues whenever the transmitter is enabled; each transmitted character aligns to the next edge of the transmit bit clock.

In general, configuring OSR for a higher oversampling ratio and/or sampling on both edges of the clock slightly improves the LPUART tolerance to baud rate mismatch between the received data and the LPUART configured baud rate. However, the three data samples in each bit will also be closer together which may impact noise sensitivity.

73.3.4 Transmitter functional description

This section describes the overall block diagram for the LPUART transmitter, as well as specialized functions for sending break and idle characters.

The transmitter output (TXD) idle state defaults to logic high; the transmitter output is inverted by setting CTRL[TXINV]. CTRL[TXINV] is cleared following reset. The transmitter is enabled by setting the CTRL[TE] bit. This queues a preamble character that is one full character frame of the idle state. The transmitter then remains idle until data is available in the transmit FIFO. Programs store data into the transmit FIFO by writing to the DATA register.

The central element of the LPUART transmitter is the transmit shift register that is 9-bit to 13-bit long depending on the setting in the CTRL[M], CTRL[M7], BAUD[M10] and BAUD[SBNS] control bits. For the remainder of this section, assume CTRL[M], CTRL[M7], BAUD[M10] and BAUD[SBNS] are cleared, selecting the normal 8-bit data mode. In 8-bit data mode, the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new character, the value waiting in the transmit data register is transferred to the transmit shift register, synchronized with the baud rate clock, and the transmit data register empty (STAT[TDRE]) status flag is set to indicate another character may be written to the transmit FIFO at the DATA register.

If no new character is waiting in the transmit FIFO after a stop bit is shifted out of the TXD pin, the transmitter sets the transmit complete flag and enters an idle mode, with TXD high, waiting for more characters to transmit.

Writing 0 to CTRL[TE] does not immediately disable the transmitter. The current transmit activity in progress must first be completed (that could include a data character, idle character or break character), although the transmitter does not start transmitting another character.

73.3.4.1 Send break and queued idle

The CTRL[SBK] bit sends break characters originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0, 9-bit to 12-bit times including the start and stop bits. A longer break of 13-bit times can be enabled by setting STAT[BRK13]. Normally, a program waits for STAT[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, write 1, and then write 0 to the CTRL[SBK] bit. This action queues a break character to be sent as soon as the shifter is available. If CTRL[SBK] remains 1 when the queued break moves into the shifter, synchronized to the baud rate clock, an additional break character is queued. When the LPUART is the receiving device, a break character is received as 0s in all data bits and a framing error (STAT[FE] = 1) is detected.

A break character can also be transmitted by writing to the DATA register with DATA[FRETSC] set and the data bits clear. This supports transmitting the break character as part of the normal data stream and also allows the DMA to transmit a break character.

When idle-line wakeup is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program waits for STAT[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, write 0, and then write 1 to the CTRL[TE] bit. This action queues an idle character to be sent as soon as the shifter is available. As long as the character in the shifter does not finish while CTRL[TE] is cleared, the LPUART transmitter never actually releases control of the TXD pin.

An idle character can also be transmitted by writing to the DATA register with DATA[FRETSC] and DATA[T9] set, and all other bits cleared. This supports transmitting the idle character as part of the normal data stream and also allows the DMA to transmit a idle character.

The length of the break character is affected by the STAT[BRK13], CTRL[M], CTRL[M7], BAUD[M10] and BAUD[SNBS] bits as shown below.

Table 534. Break character length

BRK13	M	M10	M7	SBNS	Break character length
0	0	0	0	0	10 bit times
0	0	0	0	1	11 bit times
0	0	0	1	0	9 bit times
0	0	0	1	1	10 bit times
0	1	0	X	0	11 bit times
0	1	0	X	1	12 bit times
0	X	1	X	0	12 bit times
0	X	1	X	1	13 bit times
1	0	0	0	0	13 bit times
1	0	0	0	1	13 bit times
1	0	0	1	0	12 bit times
1	0	0	1	1	12 bit times
1	1	0	X	0	14 bit times
1	1	0	X	1	14 bit times
1	X	1	X	0	15 bit times
1	X	1	X	1	15 bit times

73.3.4.2 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of CTS_B. If the clear-to-send operation is enabled, the character is transmitted when CTS_B is asserted. If CTS_B is deasserted in the middle of a transmission with characters remaining in the transmitter FIFO, the character in the transmit shift register will be complete; any characters in the FIFO will wait for the CTS_B to assert again and TXD remains in the mark state (idle state) until CTS_B is reasserted. The CTS_B pin must assert for longer than one bit period to guarantee a new transmission is started when the transmitter is idle with data to send.

If the clear-to-send operation is disabled, the transmitter ignores the state of CTS_B.

The transmitter's CTS_B signal can be enabled even if the same LPUART receiver's RTS_B signal is disabled.

73.3.4.3 Transceiver driver enable

The transmitter can use RTS_B as an enable signal for the driver of an external transceiver. See Transceiver driver enable using RTS_B for details. If the request-to-send operation is enabled, when a character is placed into an empty transmit shift register, RTS_B asserts one bit time before the start bit is transmitted. RTS_B remains asserted for the whole time that the transmit shift register has any characters. RTS_B deasserts one bit time after all characters in the transmit FIFO and shift register are completely sent, including the last stop bit. In other words, when RTS_B is used as a transceiver enable, the RTS_B asserts 1 bit time before the transmitter starts transmitting and negates 1 bit time after the transmitter goes idle.

Transmitting a break character also asserts RTS_B, with the same assertion and deassertion timing as having a character in the transmit shift register.

The transmitter's RTS_B signal asserts only when the transmitter is enabled. However, the transmitter's RTS_B signal is unaffected by its CTS_B signal. RTS_B remains asserted until the transfer is completed, even if the transmitter is disabled mid-way through a data transfer.

73.3.4.4 Transceiver driver enable using RTS_B

RS-485 is a multiple drop communication protocol in which the LPUART transceiver's driver is 3-stated unless the LPUART is driving. The RTS_B signal can be used by the transmitter to enable the driver of a transceiver. The polarity of RTS_B can be matched to the polarity of the transceiver's driver enable signal.

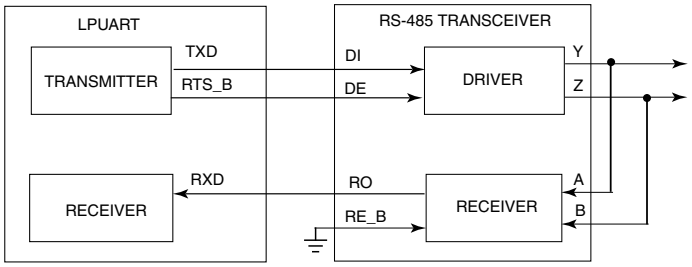


Figure 428. Transceiver driver enable using RTS_B

In the figure, the receiver enable signal is asserted. Another option for this connection is to connect RTS_B to both DE and RE_B. The transceiver's receiver is disabled while driving. A pullup can pull RXD to a non-floating value during this time. This option can be refined further by operating the LPUART in single wire mode, freeing the RXD pin for other uses.

73.3.5 Receiver functional description

In this section, the receiver block diagram is a guide for the overall receiver functional description. Next, the data sampling technique used to reconstruct receiver data is described in more detail. Finally, different variations of the receiver wakeup function are explained.

The receiver input is inverted by setting STAT[RXINV]. The receiver is enabled by setting the CTRL[RE] bit. Character frames consist of a start bit of logic 0, seven to ten data bits (msb or lsb first), and one or two stop bits of logic 1. For information about 7-bit, 9-bit or 10-bit data mode, refer to Data Modes. For the remainder of this discussion, assume the LPUART is configured for normal 8-bit data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full, the data character is transferred to the receive data register and the receive data register full (STAT[RDRF]) status flag is set. If [RDRF] was already set indicating the receive data register (buffer) was already full, the overrun (OR) status flag is set and the new data is lost.

Since the LPUART receiver is separate from the receive FIFO, the receive shift register can be receiving the next word when the receive FIFO is full and it is only at the end of the character that the next data is written into the receive FIFO, potentially triggering the overrun flag if FIFO is full.

When a program detects that the receive data register is full (STAT[RDRF] = 1), it gets the data from the receive data register by reading the DATA register. Refer to Interrupts and status flags for details about flag clearing.

73.3.5.1 Data sampling technique

The LPUART receiver supports a configurable oversampling rate of between 4× and 32× of the baud rate clock for sampling. The receiver starts by taking logic level samples at the oversampling rate times the baud rate to search for a falling edge on the RXD serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The oversampling baud rate clock divides the bit time into 4 to 32 segments from 1 to OSR (where OSR is the configured oversampling ratio). When a falling edge is located, three more samples are taken at (OSR/2), (OSR/2)+1, and (OSR/2)+2 to make sure this was a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes it is synchronized to a received character. If another falling edge is detected before the receiver is considered synchronized, the receiver restarts the sampling from the first segment.

The receiver then samples each bit time, including the start and stop bits, at (OSR/2), (OSR/2)+1, and (OSR/2)+2 to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. If any sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, the noise flag (STAT[NF]) is set when the received character is transferred to the receive FIFO.

When the LPUART receiver is configured to sample on both edges of the baud rate clock (that is, when BAUD[BOTHEDGE]=1), the number of segments in each received bit is effectively doubled (from 1 to OSR×2). The start and data bits are then sampled at OSR, OSR+1 and OSR+2. Sampling on both edges of the clock must be enabled for oversampling rates of 4× to 7× and is optional for higher oversampling rates.

The "synchronization" is a feature of the LPUART module to synchronize the internal oversampling counter with detected falling edge on Rx signal, and to adjust the data sampling window. The falling edge detection needs three consecutive '1' prior to '1'->'0' transition. After the initial falling edge detection for the start bit, the circuit continuously monitors the next falling edge, and resets the counter once another falling edge is detected. This is called "resynchronization".

- When LPUART_BAUD[RESYNCDIS] = 0, this falling edge detection and resynchronization is performed not only for the start bit but also for the rest of character reception after the start bit.
- When LPUART_BAUD[RESYNCDIS] = 1, the falling edge detection and resynchronization is performed only for the start bit. The use case for disabling the resynchronization is protocols that require this (for example, LIN 2.1 prohibits resynchronization within a byte).

  See the following table and figure.

Table 535. LPUART resynchronization

Resynchronization	LPUART_BAUD[RESYNCDIS]=0	LPUART_BAUD[RESYNCDIS]=1
For the starting bit falling edge	Yes	Yes
For every falling edges after the start bit	Yes	No

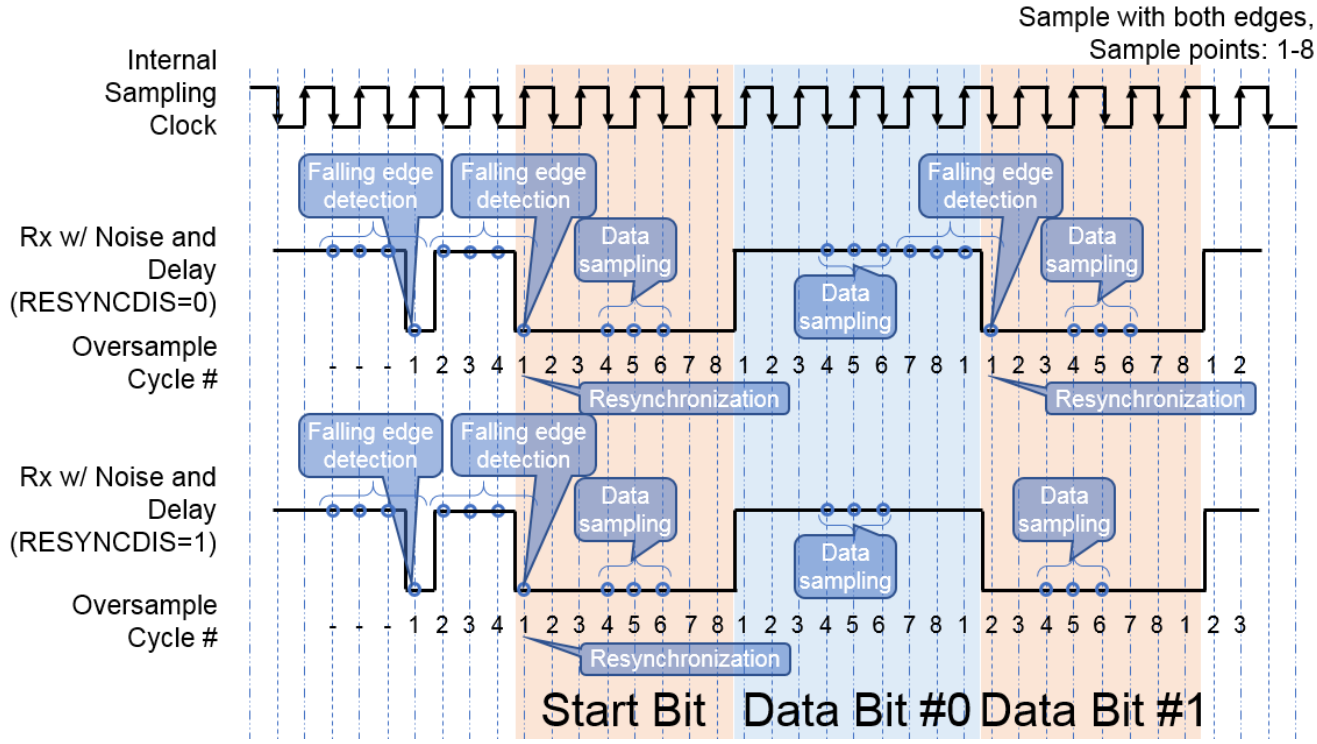


Figure 429. LPUART resynchronization

73.3.5.2 Receiver wakeup operation

Receiver wakeup and receiver address matching is a hardware mechanism that allows an LPUART receiver to ignore the characters in a message intended for a different receiver.

During receiver wakeup, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write logic 1 to the receiver wake up control bit (CTRL[RWU]). When CTRL[RWU] and STAT[RWUID] bit are set, the status flags associated with the receiver, with the exception of the idle bit, STAT[IDLE], are inhibited from setting, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, all receivers automatically force CTRL[RWU] to 0 so all receivers wake up in time to look at the first character(s) of the next message.

During receiver address matching, the address matching is performed in hardware and the LPUART receiver ignores all characters that do not meet the address match requirements.

Table 536. Receiver Wakeup Options

RWU	MA1 | MA2	MATCFG	WAKE:RWUID	Receiver Wakeup
0	0	X	X	Normal operation
1	0	00	00	Receiver wakeup on idle line, IDLE flag not set
1	0	00	01	Receiver wakeup on idle line, IDLE flag set
1	0	00	10	Receiver wakeup on address mark

Table continues on the next page...

Table 536. Receiver Wakeup Options (continued)

RWU	MA1 | MA2	MATCFG	WAKE:RWUID	Receiver Wakeup
1	1	11	10	Receiver wakeup on data match
0	1	00	X0	Address mark address match, IDLE flag not set for discarded characters
0	1	00	X1	Address mark address match, IDLE flag set for discarded characters
0	1	01	X0	Idle line address match
0	1	10	X0	Match On and Match Off, IDLE flag not set for discarded characters
0	1	10	X1	Match On and Match Off, IDLE flag set for discarded characters

73.3.5.2.1 Idle-line wakeup

When CTRL[WAKE] is cleared, the receiver is configured for idle-line wakeup. In this mode, CTRL[RWU] is cleared automatically when the receiver detects a full character time of the idle-line level. The CTRL[M], CTRL[M7], and BAUD[M10] control bits select 7-bit to 10-bit data mode and the BAUD[SBNS] bit selects 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 9 to 13 bit times because of the start and stop bits.

When CTRL[RWU] is 1 and STAT[RWUID] is 0, the idle condition that wakes up the receiver does not set the STAT[IDLE] flag. The receiver wakes up and waits for the first data character of the next message that sets the STAT[RDRF] flag and generates an interrupt if enabled. When STAT[RWUID] is 1, any idle condition sets the STAT[IDLE] flag and generates an interrupt if enabled, regardless of whether CTRL[RWU] is 0 or 1.

The idle-line type (CTRL[ILT]) control bit selects one of two ways to detect an idle line.

- When CTRL[ILT] is cleared, the idle bit counter starts after the start bit so the stop bit and any logic 1s at the end of a character count toward the full character time of idle.
- When CTRL[ILT] is set, the idle bit counter does not start until after the stop bit time, so the idle detection is not affected by the data in the last character of the previous message.

73.3.5.2.2 Address-mark wakeup

When CTRL[WAKE] is set, the receiver is configured for address-mark wakeup. In this mode, CTRL[RWU] is cleared automatically when the receiver detects a logic 1 in the most significant bit of the received character. When parity is enabled, the second most significant bit is used for address-mark wakeup.

Address-mark wakeup allows messages to contain idle characters, but requires one bit be reserved for use in address frames. The logic 1 in the most significant bit (or second most significant bit when parity is enabled) of an address frame clears the CTRL[RWU] bit and sets the STAT[RDRF] flag. In this case, the character with the address-mark bit is received even though the receiver was sleeping during most of this character time.

73.3.5.2.3 Data match wakeup

When CTRL[RWU] is set, CTRL[WAKE] is set and BAUD[MATCFG] equals 11, the receiver is configured for data match wakeup. In this mode, CTRL[RWU] is cleared automatically when the receiver detects a character that matches MATCH[MA1] field when BAUD[MAEN1] is set, or that matches MATCH[MA2] when BAUD[MAEN2] is set.

73.3.5.2.4 Address Match operation

Address match operation is enabled when the BAUD[MAEN1] or BAUD[MAEN2] bit is set and BAUD[MATCFG] is equal to 00. In this function, a character received by the RXD pin with a logic 1 in the most significant bit (or second most significant bit when parity is enabled) is considered an address and is compared with the associated MATCH[MA1] or MATCH[MA2] field. The character is only transferred to the receive buffer, and STAT[RDRF] is set, if the comparison matches. All subsequent characters received with a logic 0 in the most significant bit (or second most significant bit when parity is enabled) are considered to be data associated with the address and are transferred to the receive FIFO. If no marked address match occurs then no transfer is made to the receive FIFO, and all following characters with logic zero in the most significant bit (or second most significant bit when parity is enabled) are also discarded. If both the BAUD[MAEN1] and BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive FIFO.

Address match operation functions in the same way for both MATCH[MA1] and MATCH[MA2] fields.

- If only one of BAUD[MAEN1] and BAUD[MAEN2] is asserted, a marked address is compared only with the associated match register field and data is transferred to the receive FIFO only on a match.
- If BAUD[MAEN1] and BAUD[MAEN2] are asserted, a marked address is compared with both match registers and data is transferred only on a match with either of the MATCH[MA1] or MATCH[MA2] fields.

73.3.5.2.5 Idle Match operation

Idle match operation is enabled when the BAUD[MAEN1] or BAUD[MAEN2] bit is set and BAUD[MATCFG] is equal to 01. In this function, the first character received by the RXD pin after an idle line condition is considered an address and is compared with the associated MATCH[MA1] or MATCH[MA2] field. The character is transferred only to the receive buffer, and STAT[RDRF] is set, if the comparison matches. All subsequent characters are considered to be data associated with the address and are transferred to the receive FIFO until the next idle line condition is detected. If no address match occurs then no transfer is made to the receive FIFO, and all following frames until the next idle condition are also discarded. If both the BAUD[MAEN1] and BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive FIFO.

Idle match operation functions in the same way for both MATCH[MA1] or MATCH[MA2] fields.

- If only one of BAUD[MAEN1] and BAUD[MAEN2] is asserted, the first character after an idle line is compared only with the associated match register and data is transferred to the receive FIFO only on a match.
- If BAUD[MAEN1] and BAUD[MAEN2] are asserted, the first character after an idle line is compared with both MATCH[MA1] or MATCH[MA2] fields and data is transferred only on a match with either of the fields.

73.3.5.2.6 Match On Match Off operation

Match on, match off operation is enabled when both BAUD[MAEN1] and BAUD[MAEN2] are set and BAUD[MATCFG] is equal to 10. In this function, a character received by the RXD pin that matches MATCH[MA1] is received and transferred to the receive buffer, and STAT[RDRF] is set. All subsequent characters are considered to be data and are also transferred to the receive FIFO, until a character is received that matches MATCH[MA2] field. The character that matches MATCH[MA2] and all following characters are discarded; this continues until another character that matches MATCH[MA1] is received. If both the BAUD[MAEN1] and BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive FIFO.

NOTE

Match on, match off operation requires both BAUD[MAEN1] and BAUD[MAEN2] to be asserted.

73.3.5.3 Hardware flow control

To support hardware flow control, the receiver can be programmed to automatically deassert and assert RTS_B.

- RTS_B remains asserted until the transfer is complete, even if the transmitter is disabled midway through a data transfer. See Transceiver driver enable using RTS_B for more details.
- If the receiver request-to-send functionality is enabled, the receiver automatically deasserts RTS_B if the number of characters in the receiver data register is full or a start bit is detected that causes the receiver data register to be full.
- The receiver asserts RTS_B when the number of characters in the receiver data register is not full and has not detected a start bit that causes the receiver data register to be full. It is not affected if STAT[RDRF] is asserted.
- Even if RTS_B is deasserted, the receiver continues to receive characters until the receive FIFO is overrun.
- If the receiver request-to-send functionality is disabled, the receiver RTS_B remains deasserted.

73.3.6 Additional LPUART functions

The following sections describe additional LPUART functions.

73.3.6.1 Data Modes

The LPUART transmitter and receiver can be configured to operate in 7-bit data mode by setting CTRL[M7], 9-bit data mode by setting the CTRL[M] or 10-bit data mode by setting BAUD[M10]. In 9-bit mode, there is a ninth data bit and in 10-bit mode, there is a tenth data bit.

When performing 8-bit writes to the transmit FIFO, the 9th and 10th bit are pushed into the FIFO from CTRL[T8] and CTRL[T9]. For coherent 8-bit writes, write to CTRL[T8] and CTRL[T9] before writing to DATA[7:0], but if values in CTRL[T8] or CTRL[T9] do not need to change then it is not necessary to update CTRL[T8] and CTRL[T9] before every 8-bit write to the DATA register.

When performing 16-bit or 32-bit writes to the transmit FIFO, all 10-bits are pushed into the transmit FIFO from the write data.

When performing 8-bit reads of the receive FIFO, the 9th and 10th bit are held in CTRL[R8] and CTRL[R9] but should be read before reading the DATA register. A 16-bit or 32-bit read of the receive FIFO will return all 10-bits in the DATA register.

The 9-bit data mode is typically used with parity to allow eight bits of data plus the parity in the ninth bit, or it is used with address-mark wakeup so the ninth data bit can serve as the wakeup bit. The 10-bit data mode is typically used with parity and address-mark wakeup so the ninth data bit can serve as the wakeup bit and the tenth bit as the parity bit. In custom protocols, the ninth and/or tenth bits can also serve as software-controlled markers.

73.3.6.2 Idle length

An idle character is a character where the start bit, all data bits and stop bits are in the mark position (idle state, generally logic 1). The CTRL[ILT] bit can be configured to start detecting an idle character from the previous start bit (any data bits and stop bits count towards the idle character detection) or from the previous stop bit.

The number of idle characters that must be received before an idle line condition is detected can also be configured using the CTRL[IDLECFG] field. This field configures the number of idle characters that must be received before the STAT[IDLE] flag is set, the STAT[RAF] flag is cleared and the DATA[IDLINE] flag is set with the next received character.

Idle-line wakeup and idle match operation are also affected by the CTRL[IDLECFG] field. When address match or match on/off operation is enabled, setting the STAT[RWUID] bit causes any discarded characters to be treated as if they were idle characters.

73.3.6.3 Loop mode

When CTRL[LOOPS] is set, the CTRL[RSRC] bit chooses between loop mode (CTRL[RSRC] = 0) or single-wire mode (CTRL[RSRC] = 1). Loop mode is sometimes used to check software, independent of connections in the external system, to help isolate system problems. In this mode, the transmitter output is internally connected to the receiver input and the RXD pin is not used by the LPUART.

73.3.6.4 Single-wire operation

When CTRL[LOOPS] is set, the CTRL[RSRC] bit chooses between loop mode (CTRL[RSRC] = 0) or single-wire mode (CTRL[RSRC] = 1). Single-wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the TXD pin (the RXD pin is not used).

In single-wire mode, the CTRL[TXDIR] bit controls the direction of serial data on the TXD pin. When CTRL[TXDIR] is cleared, the TXD pin is an input to the receiver and the transmitter is temporarily disconnected from the TXD pin so an external device can send serial data to the receiver. When CTRL[TXDIR] is set, the TXD pin is an output driven by the transmitter, the internal loop back connection is disabled, and as a result the receiver cannot receive characters that are sent out by the transmitter.

73.3.7 Infrared interface

The LPUART provides the capability of transmitting narrow pulses to an IR LED and receiving narrow pulses and transforming them to serial bits, which are sent to the LPUART. The IrDA physical layer specification defines a half-duplex infrared communication link for exchanging data. The full standard includes data rates up to 16 Mbits/s. The LPUART IrDA support is limited to SIR mode which supports data rates only between 2.4 kbits/s and 115.2 kbits/s.

The LPUART has an infrared transmit encoder and receive decoder. The infrared decoder converts the received character from the IrDA format to the NRZ format used by the receiver. It also has a OSR oversampling baud rate clock counter that filters noise and indicates when a 1 is received. The LPUART transmits serial bits of data that are encoded by the infrared submodule to transmit a narrow pulse for every zero bit. No pulse is transmitted for every one bit. When receiving data, the IR pulses are detected using an IR photo diode (external to LPUART) and transformed to CMOS levels by the IR receive decoder. The narrow pulses are then stretched by the infrared receive decoder to get back to a serial bit stream to be received by the LPUART. The polarity of transmitted pulses and expected receive pulses can be inverted so that a direct connection can be made to external IrDA transceiver modules that use active high pulses.

The infrared submodule receives its clock sources from the LPUART. One of these two clocks are selected in the infrared submodule to generate either 1/OSR, 2/OSR, 3/OSR, or 4/OSR narrow pulses during transmission.

73.3.7.1 Infrared transmit encoder

The infrared transmit encoder converts serial bits of data from transmit shift register to the TXD signal. A narrow pulse is transmitted for a 0 bit and no pulse for a 1 bit. The narrow pulse is sent at the start of the bit with a duration of 1/OSR, 2/OSR, 3/OSR, or 4/OSR of a bit time. A narrow low pulse is transmitted for a 0 bit when CTRL[TXINV] is cleared, while a narrow high pulse is transmitted for a 0 bit when CTRL[TXINV] is set.

73.3.7.2 Infrared receive decoder

The infrared receive block converts data from the RXD signal to the receive shift register. A narrow pulse is expected for each 0 received and no pulse is expected for each 1 received. A narrow low pulse is expected for a 0 bit when STAT[RXINV] is cleared, while a narrow high pulse is expected for a 0 bit when STAT[RXINV] is set. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

73.3.7.3 Start bit detection

When STAT[RXINV] is cleared, the first falling edge of the received character corresponds to the start bit. The infrared decoder resets its counter. At this time, the receiver also begins its start bit detection process. After the start bit is detected, the receiver synchronizes its bit times to this start bit time. For the rest of the character reception, the infrared decoder's counter and the receiver's bit time counter count independently from each other.

73.3.7.4 Noise filtering

Any further rising edges detected during the first half of the infrared decoder counter are ignored by the decoder. Any pulses less than one oversampling baud clock can be undetected by it regardless of whether it is seen in the first or second half of the count.

73.3.7.5 Low-bit detection

During the second half of the decoder count, a rising edge is decoded as a 0, which is sent to the receiver. The decoder counter is also reset.

73.3.7.6 High-bit detection

At OSR oversampling baud rate clocks after the previous rising edge, if a rising edge is not seen, then the decoder sends a 1 to the receiver.

If the next bit is a 0, which arrives late, then a low-bit is detected according to Low-bit detection. The value sent to the receiver is changed from 1 to a 0. Then, if a noise pulse occurs outside the receiver's bit time sampling period, then the delay of a 0 is not recorded as noise.

73.3.8 Interrupts and status flags

The LPUART transmitter has two status flags that can optionally generate hardware interrupt requests. Transmit Data Register Empty (STAT[TDRE]) indicates when there is room in the transmit FIFO to write another transmit character to the DATA register. If the Transmit Interrupt Enable (CTRL[TIE]) bit is set, a hardware interrupt is requested when STAT[TDRE] is set. Transmit complete (STAT[TC]) indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with TXD at the inactive level. This flag is often used in systems with modems to determine when it is safe to turn off the modem. If the transmit complete interrupt enable (CTRL[TCIE]) bit is set, a hardware interrupt is requested when STAT[TC] is set. Instead of hardware interrupts, software polling may be used to monitor the STAT[TDRE] and STAT[TC] status flags if the corresponding CTRL[TIE] or CTRL[TCIE] local interrupt masks are cleared.

When a program detects that the receive data register is full (STAT[RDRF] = 1), it gets the data from the receive data register by reading the DATA register. The STAT[RDRF] flag is cleared by reading the DATA register.

The IDLE status flag includes logic that prevents it from getting set repeatedly when the RXD line remains idle for an extended period of time. IDLE is cleared by writing 1 to the STAT[IDLE] flag. After STAT[IDLE] has been cleared, it cannot become set again until the receiver has received at least one new character and has set STAT[RDRF].

If the associated error is detected in the received character that caused STAT[RDRF] to be set, the error flags — noise flag (STAT[NF]), framing error (STAT[FE]), and parity error flag (STAT[PF]) — are set at the same time as STAT[RDRF]. These flags are not set in overrun cases.

If STAT[RDRF] is already set when a new character is ready to be transferred from the receive shifter to the receive FIFO, the overrun (STAT[OR]) flag is set instead of the data along with any associated STAT[NF], STAT[FE], or STAT[PF] condition is lost.

If the received character matches the contents of MATCH[MA1] and/or MATCH[MA2] then the STAT[MA1F] and/or STAT[MA2F] flags are set at the same time that STAT[RDRF] is set.

At any time, an active edge on the RXD serial data input pin causes the STAT[RXEDGIF] flag to set. The STAT[RXEDGIF] flag is cleared by writing a 1 to it. This function depends on the receiver being enabled (CTRL[RE] = 1).

73.3.9 Peripheral Triggers

The connection of the LPUART peripheral triggers with other peripherals are device specific.

Output Triggers

The LPUART generates the following output triggers that can be connected to other peripherals on the device.

- The transmit word trigger asserts at the end of each transmitted word, it negates after one bit period.
- The transmit data trigger is identical to the TXD pin output, but without support for input trigger modulation.
- The receive word trigger asserts at the end of each received word that is written to the receive FIFO, for one oversampling clock period.
- The receive idle trigger asserts when the idle flag would set, for one oversampling clock period.

Input Trigger

The LPUART supports one peripheral input trigger, that can be configured in one of the following ways.

- The CTS function must be enabled. The input trigger can be connected in place of the CTS_B pin input. The input trigger must assert for longer than one bit clock period when the transmitter is idle with data to send to guarantee a new transmission.
- The input trigger can modulate the transmit data output (trigger is logically ANDed with the TXD output). The input trigger is expected to be a free running clock (carrier signal) generated from a timer or PWM source with a frequency that is greater than the bit clock frequency. The carrier signal must not toggle faster than the maximum supported bit time.
- The input trigger can be connected in place of the RXD pin input. The input trigger is expected to be generated from a receive data source, such as analog comparator or external pin.

73.4 Clocking and Resets

Table 537. Clocks

LPUART Functional clock	The LPUART functional clock is asynchronous to the bus clock and can remain enabled in low power modes to support transmit and/or receive, including low power wakeups.
Bus clock	The bus clock is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPUART transmit and receive registers, including the FIFOs.

Table 538. Resets

Chip reset	The logic and registers for the LPUART transmitter and receiver are reset to their default state on a chip reset.
Software reset	Resets the LPUART logic and registers to their default state, except for the Global Register. The LPUART software reset is in the Global Register GLOBAL[RST].
FIFO reset	The LPUART implements write-only control bits that reset the transmit FIFO (FIFO[TXFLUSH]) and receive FIFO (FIFO[RXFLUSH]). After a FIFO is reset, that FIFO is empty.

73.5 External signals

Table 539.

Signal	Description	I/O
TXD	Transmit data. This pin is normally an output, but is an input (tristated) in single wire mode whenever the transmitter is disabled or transmit direction is configured for receive data.	I/O
RXD	Receive data.	I
CTS_B	Clear to send.	I
RTS_B	Request to send.	O

73.6 Register definition

The LPUART includes registers to control baud rate, select options, report status, and store transmit/receive data. Access to an address outside the valid memory map generates a bus error.

NOTE

Writing a Read-Only (RO) register or reading a Write-Only (WO) register can cause bus errors. This module does not check if programmed values in the registers are correct; the application software must ensure that valid programmed values are being written.

73.6.1 LPUART register descriptions

73.6.1.1 LPUART memory map

LPUART_0 base address: 4032_8000h

LPUART_1 base address: 4032_C000h

LPUART_2 base address: 4033_0000h

LPUART_3 base address: 4033_4000h

LPUART_4 base address: 4033_8000h

LPUART_5 base address: 4033_C000h

LPUART_6 base address: 4034_0000h

LPUART_7 base address: 4034_4000h

LPUART_8 base address: 4048_C000h

LPUART_9 base address: 4049_0000h

LPUART_10 base address: 4049_4000h

LPUART_11 base address: 4049_8000h

LPUART_12 base address: 4049_C000h

LPUART_13 base address: 404A_0000h

LPUART_14 base address: 404A_4000h

LPUART_15 base address: 404A_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	R	0403_0003h
4h	Parameter Register (PARAM)	32	R	0000_0202h
8h	LPUART Global Register (GLOBAL)	32	RW	0000_0000h
Ch	LPUART Pin Configuration Register (PINCFG)	32	RW	0000_0000h
10h	LPUART Baud Rate Register (BAUD)	32	RW	0F00_0004h
14h	LPUART Status Register (STAT)	32	RW	00C0_0000h
18h	LPUART Control Register (CTRL)	32	RW	0000_0000h
1Ch	LPUART Data Register (DATA)	32	RW	0000_1000h
20h	LPUART Match Address Register (MATCH)	32	RW	0000_0000h
24h	LPUART Modem IrDA Register (MODIR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
28h	LPUART FIFO Register (FIFO)	32	RW	00C0_0011h
2Ch	LPUART Watermark Register (WATER)	32	RW	0000_0000h
30h	Data read-only Register (DATARO)	32	R	0000_1000h

73.6.1.2 Version ID Register (VERID)

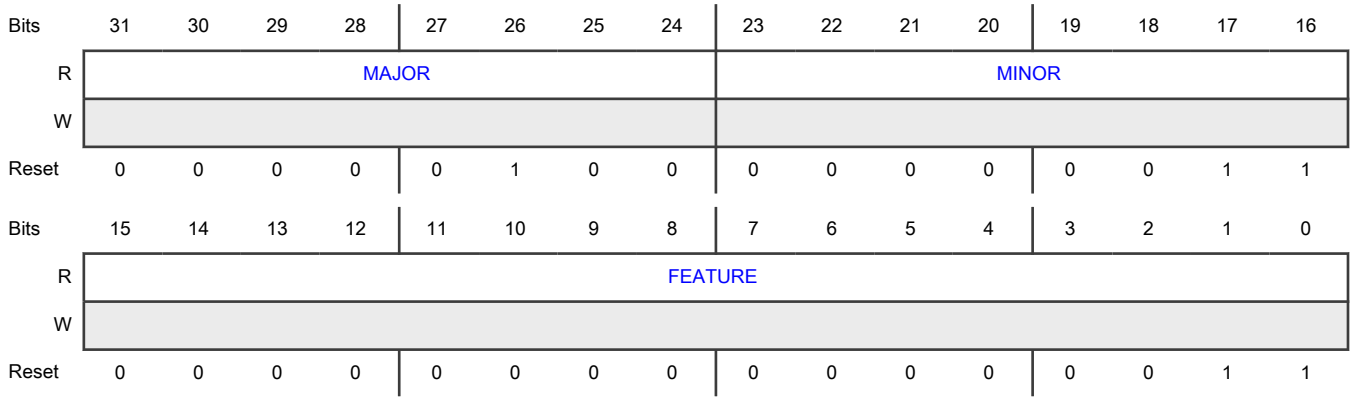
Offset

Register	Offset
VERID	0h

Function

The Version ID register indicates the version integrated for this instance on the device and also indicates inclusion/exclusion of several optional features.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number This read-only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read-only field returns the minor version number for the module specification.
15-0	Feature Identification Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
FEATURE	This read-only field returns the feature set number. 0000_0000_0000_0001b - Standard feature set. 0000_0000_0000_0011b - Standard feature set with MODEM/IrDA support.

73.6.1.3 Parameter Register (PARAM)

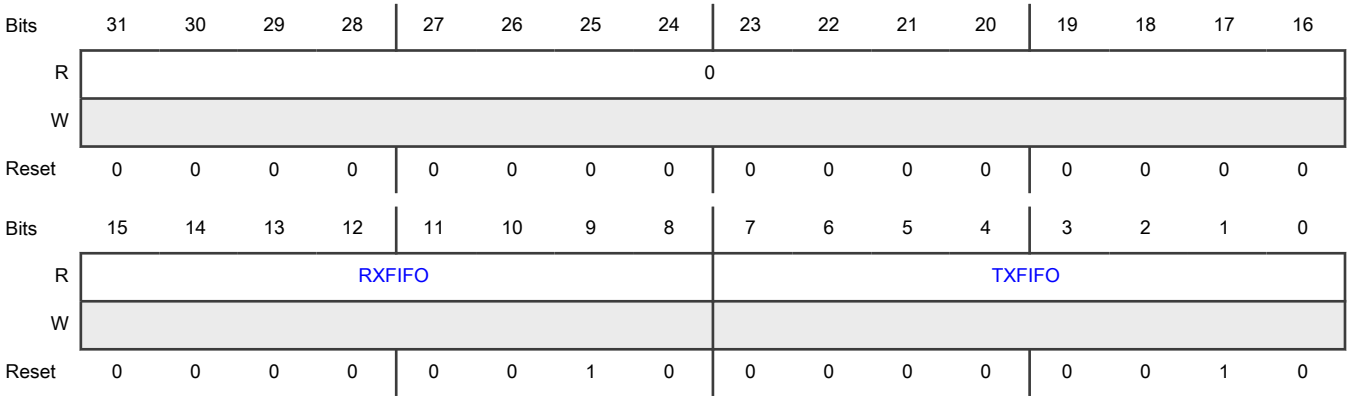
Offset

Register	Offset
PARAM	4h

Function

The Parameter register indicates the parameter configuration for this instance on the device.

Diagram



Fields

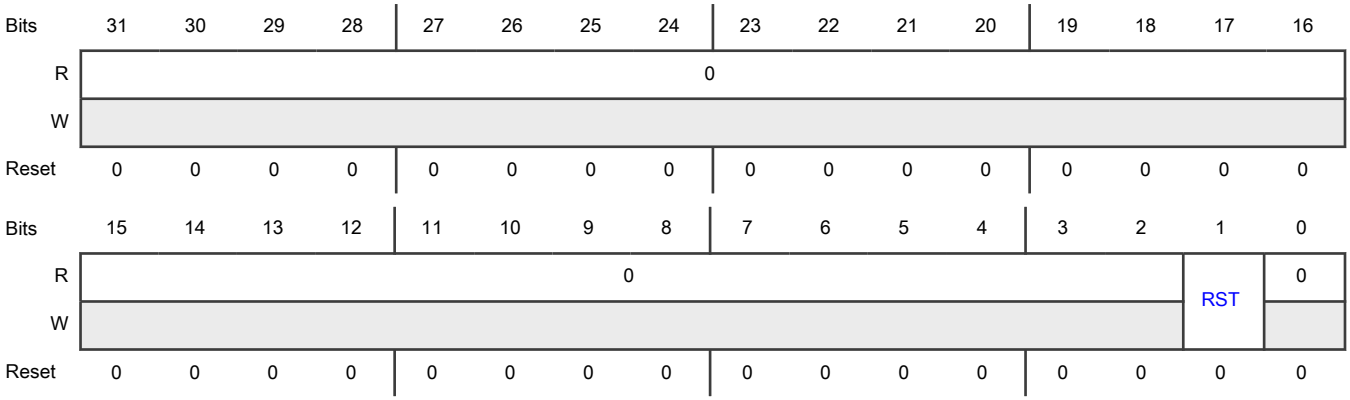
Field	Function
31-16 —	Reserved
15-8 RXFIFO	Receive FIFO Size The number of characters in the receive FIFO is 2^RXFIFO.
7-0 TXFIFO	Transmit FIFO Size The number of characters in the transmit FIFO is 2^TXFIFO.

73.6.1.4 LPUART Global Register (GLOBAL)

Offset

Register	Offset
GLOBAL	8h

Diagram



Fields

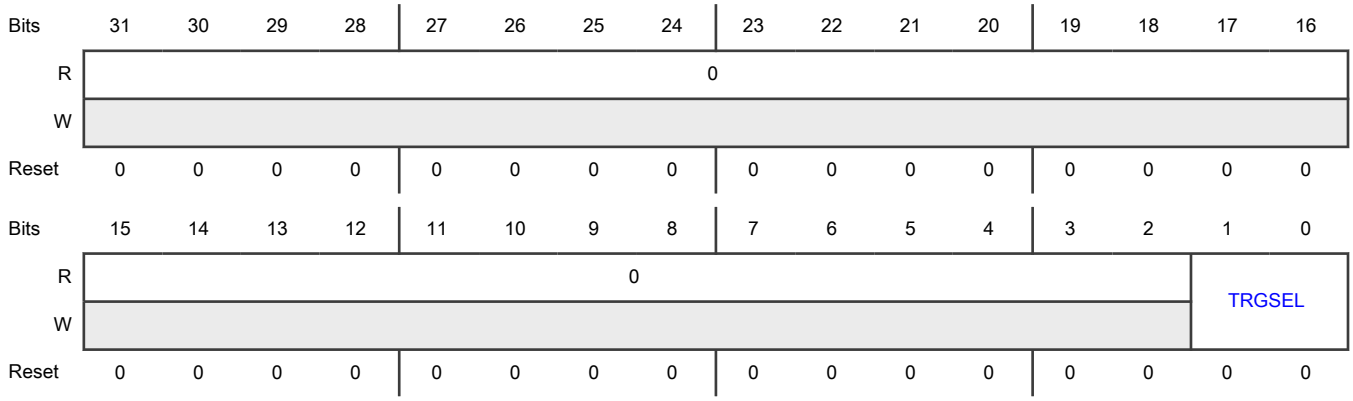
Field	Function
31-2 —	Reserved
1 RST	Software Reset Resets all internal logic and registers, except the Global Register. The reset takes effect immediately and remains asserted until negated by software. There is no minimum delay required before clearing the software reset. 0b - Module is not reset. 1b - Module is reset.
0 —	Reserved

73.6.1.5 LPUART Pin Configuration Register (PINCFG)

Offset

Register	Offset
PINCFG	Ch

Diagram



Fields

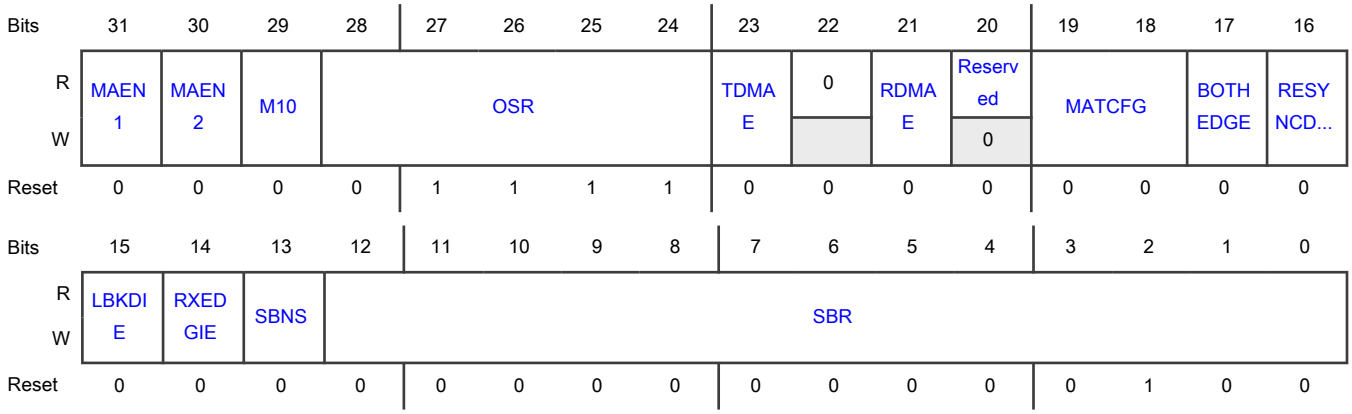
Field	Function
31-2 —	Reserved
1-0 TRGSEL	Trigger Select Configures the input trigger usage. This field should be changed only when the transmitter and receiver are both disabled. 00b - Input trigger is disabled. 01b - Input trigger is used instead of RXD pin input. 10b - Input trigger is used instead of CTS_B pin input. 11b - Input trigger is used to modulate the TXD pin output. The TXD pin output (after TXINV configuration) is internally ANDed with the input trigger.

73.6.1.6 LPUART Baud Rate Register (BAUD)

Offset

Register	Offset
BAUD	10h

Diagram



Fields

Field	Function
31 MAEN1	Match Address Mode Enable 1 0b - Normal operation. 1b - Enables automatic address matching or data matching mode for MATCH[MA1].
30 MAEN2	Match Address Mode Enable 2 0b - Normal operation. 1b - Enables automatic address matching or data matching mode for MATCH[MA2].
29 M10	10-bit Mode select The M10 bit causes a tenth bit to be part of the serial transmission. This bit should be changed only when the transmitter and receiver are both disabled. 0b - Receiver and transmitter use 7-bit to 9-bit data characters. 1b - Receiver and transmitter use 10-bit data characters.
28-24 OSR	Oversampling Ratio This field configures the oversampling ratio for the receiver. This field should be changed only when the transmitter and receiver are both disabled. 0_0000b - Writing 0 to this field results in an oversampling ratio of 16 0_0001b - Reserved 0_0010b - Reserved 0_0011b - Oversampling ratio of 4, requires BOTHEDGE to be set. 0_0100b - Oversampling ratio of 5, requires BOTHEDGE to be set. 0_0101b - Oversampling ratio of 6, requires BOTHEDGE to be set. 0_0110b - Oversampling ratio of 7, requires BOTHEDGE to be set. 0_0111b - Oversampling ratio of 8.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0_1000b - Oversampling ratio of 9. 0_1001b - Oversampling ratio of 10. 0_1010b - Oversampling ratio of 11. 0_1011b - Oversampling ratio of 12. 0_1100b - Oversampling ratio of 13. 0_1101b - Oversampling ratio of 14. 0_1110b - Oversampling ratio of 15. 0_1111b - Oversampling ratio of 16. 1_0000b - Oversampling ratio of 17. 1_0001b - Oversampling ratio of 18. 1_0010b - Oversampling ratio of 19. 1_0011b - Oversampling ratio of 20. 1_0100b - Oversampling ratio of 21. 1_0101b - Oversampling ratio of 22. 1_0110b - Oversampling ratio of 23. 1_0111b - Oversampling ratio of 24. 1_1000b - Oversampling ratio of 25. 1_1001b - Oversampling ratio of 26. 1_1010b - Oversampling ratio of 27. 1_1011b - Oversampling ratio of 28. 1_1100b - Oversampling ratio of 29. 1_1101b - Oversampling ratio of 30. 1_1110b - Oversampling ratio of 31. 1_1111b - Oversampling ratio of 32.
23 TDMAE	Transmitter DMA Enable TDMAE configures the transmit data register empty flag, STAT[TDRE], to generate a DMA request. 0b - DMA request disabled. 1b - DMA request enabled.
22 —	Reserved
21 RDMAE	Receiver Full DMA Enable RDMAE configures the receiver data register full flag, STAT[RDRF], to generate a DMA request.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - DMA request disabled. 1b - DMA request enabled.
20 —	Reserved
19-18 MATCFG	Match Configuration Configures the match addressing mode used. This field should be changed only when the transmitter and receiver are both disabled. 00b - Address Match Wakeup 01b - Idle Match Wakeup 10b - Match On and Match Off 11b - Enables RWU on Data Match and Match On/Off for transmitter CTS input
17 BOTHEDGE	Both Edge Sampling Enables sampling of the received data on both edges of the baud rate clock, effectively doubling the number of times the receiver samples the input data for a given oversampling ratio. This bit must be set for oversampling ratios between x4 and x7 and is optional for higher oversampling ratios. This bit should be changed only when the receiver is disabled. 0b - Receiver samples input data using the rising edge of the baud rate clock. 1b - Receiver samples input data using the rising and falling edge of the baud rate clock.
16 RESYNCDIS	Resynchronization Disable When set, this field disables the resynchronization of the received data word when a data one followed by data zero transition is detected. This bit should be changed only when the receiver is disabled. 0b - Resynchronization during received data word is supported. 1b - Resynchronization during received data word is disabled.
15 LBKDIE	LIN Break Detect Interrupt Enable LBKDIE enables the LIN break detect flag, STAT[LBKDIF], to generate interrupt requests. 0b - Hardware interrupts from STAT[LBKDIF] flag are disabled (use polling). 1b - Hardware interrupt is requested when STAT[LBKDIF] flag is 1.
14 RXEDGIE	RX Input Active Edge Interrupt Enable Enables the receive input active edge, STAT[RXEDGIF], to generate interrupt requests. Changing CTRL[LOOPS] or CTRL[RSRC] when RXEDGIE is set can cause the STAT[RXEDGIF] flag to set. 0b - Hardware interrupts from STAT[RXEDGIF] are disabled. 1b - Hardware interrupt is requested when STAT[RXEDGIF] flag is 1.
13	Stop Bit Number Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBNS	SBNS determines whether data characters have one or two stop bits. This bit should be changed only when the transmitter and receiver are both disabled. 0b - One stop bit. 1b - Two stop bits.
12-0 SBR	Baud Rate Modulo Divisor. The 13 bits in SBR[12:0] set the modulo divide rate for the baud rate generator. When SBR is 1 - 8191, the baud rate equals "baud clock / ((OSR+1) * SBR)". The 13-bit baud rate setting [SBR12:SBR0] must be updated only when the transmitter and receiver are both disabled (CTRL[RE] and CTRL[TE] are both 0).

73.6.1.7 LPUART Status Register (STAT)

Offset

Register	Offset
STAT	14h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LBKDIF	RXEDGIF	MSBF	RXINV	RWUID	BRK13	LBKDE	RAF	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W	W1C	W1C										W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MA1F	MA2F	0				0		0						AME	LBKFE
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LBKDIF	LIN Break Detect Interrupt Flag LBKDIF is set when the LIN break detect circuitry is enabled and a LIN break character is detected. Writing a 1 clears the LBKDIF field. 0b - No LIN break character has been detected. 1b - LIN break character has been detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 RXEDGIF	RXD Pin Active Edge Interrupt Flag RXEDGIF is set whenever the receiver is enabled and an active edge (falling if RXINV = 0, rising if RXINV=1,) on the RXD pin occurs. To clear RXEDGIF, write a 1 to the RXEDGIF field. 0b - No active edge on the receive pin has occurred. 1b - An active edge on the receive pin has occurred.
29 MSBF	MSB First Setting this bit reverses the order of the bits that are transmitted and received on the wire. This bit does not affect the polarity of the bits, the location of the parity bit, or the location of the start or stop bits. This bit should be changed only when the transmitter and receiver are both disabled. 0b - LSB (bit0) is the first bit that is transmitted following the start bit. Further, the first bit received after the start bit is identified as bit0. 1b - MSB (identified as bit9, bit8, bit7 or bit6) is the first bit that is transmitted following the start bit depending on the setting of CTRL[M], CTRL[PE] and BAUD[M10]. .
28 RXINV	Receive Data Inversion Setting this bit reverses the polarity of the received data input. This bit should be changed only when the receiver is disabled. **NOTE** Setting RXINV inverts the RXD input for all cases: data bits, start and stop bits, break, and idle. 0b - Receive data not inverted. 1b - Receive data inverted.
27 RWUID	Receive Wake Up Idle Detect For RWU on idle character, RWUID controls whether the idle character that wakes up the receiver sets the IDLE bit. For address match wakeup, RWUID controls if the IDLE bit is set when the address does not match. This bit should be changed only when the receiver is disabled. 0b - During receive standby state (RWU = 1), the IDLE bit does not get set upon detection of an idle character. During address match wakeup, the IDLE bit does not set when an address does not match. 1b - During receive standby state (RWU = 1), the IDLE bit gets set upon detection of an idle character. During address match wakeup, the IDLE bit does set when an address does not match.
26 BRK13	Break Character Generation Length BRK13 selects a longer transmitted break character length. The state of this bit does not affect the detection of a framing error. This bit should be changed only when the transmitter is disabled. A break character can be sent by setting CTRL[SBK] or by writing the transmit FIFO with DATA[FRETSC] set and DATA[R9T9] clear.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Break character is transmitted with length of 9 to 13 bit times. 1b - Break character is transmitted with length of 12 to 15 bit times.
25 LBKDE	LIN Break Detection Enable LBKDE selects a longer break character detection length. While LBKDE is set, receive data is not stored in the receive FIFO. **NOTE** The LBKDE bit enables the LIN break detect circuit and disables writing receive data to FIFO. So it essentially ignores all characters except a LIN break. 0b - LIN break detect is disabled, normal break character can be detected. 1b - LIN break detect is enabled. LIN break character is detected at length of 11 bit times (if M = 0) or 12 (if M = 1) or 13 (M10 = 1).
24 RAF	Receiver Active Flag RAF is set when the receiver detects the beginning of a valid start bit, and RAF is cleared automatically when the receiver detects an idle line. 0b - LPUART receiver idle waiting for a start bit. 1b - LPUART receiver active (RXD input not idle).
23 TDRE	Transmit Data Register Empty Flag When the transmit FIFO is enabled, TDRE is set when the number of datawords in the transmit FIFO is equal to or less than the number indicated by WATER[TXWATER]. To clear TDRE, write to the DATA register until the number of words in the transmit FIFO is greater than the number indicated by WATER[TXWATER]. When the transmit FIFO is disabled, TDRE is set when the transmit DATA register is empty. To clear TDRE, write to the DATA register. TDRE is not affected by a character that is in the process of being transmitted; it is updated at the start of each transmitted character. 0b - Transmit FIFO level is greater than watermark. 1b - Transmit FIFO level is equal or less than watermark.
22 TC	Transmission Complete Flag TC is cleared when there is a transmission in progress or when a preamble or break character is loaded. TC is set when the transmit buffer is empty and no data, preamble, or break character is being transmitted. When TC is set, the transmit data output signal becomes idle (logic 1). TC is cleared by writing to the DATA register to transmit new data, queuing a preamble by clearing and then setting CTRL[TE], queuing a break character by writing 1 to CTRL[SBK]. 0b - Transmitter active (sending data, a preamble, or a break). 1b - Transmitter idle (transmission activity complete).
21 RDRF	Receive Data Register Full Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When the receive FIFO is enabled, RDRF is set when the number of datawords in the receive buffer is greater than the number indicated by WATER[RXWATER]. To clear RDRF, read the DATA register until the number of datawords in the receive FIFO is equal to or less than the number indicated by WATER[RXWATER]. When the receive FIFO is disabled, RDRF is set when the receive buffer (the DATA register) is full. To clear RDRF, read the DATA register. A character that is in the process of being received does not cause a change in RDRF until the entire character is received. Even if RDRF is set, the character continues to be received until an overrun condition occurs once the entire character is received. 0b - Receive FIFO level is less than watermark. 1b - Receive FIFO level is equal or greater than watermark.
20 IDLE	Idle Line Flag IDLE is set when the LPUART receive line becomes idle for a full character time after a period of activity. When CTRL[ILT] is cleared, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bits time count toward the full character time of logic high, 10 to 13 bit times, needed for the receiver to detect an idle line. When CTRL[ILT] is set, the receiver doesn't start counting idle bit times until after the stop bits. The stop bits and any logic high bit times at the end of the previous character do not count toward the full character time of logic high needed for the receiver to detect an idle line. To clear IDLE, write logic 1 to the IDLE flag. After IDLE has been cleared, it cannot be set again until after a new character is stored in the receive buffer or a LIN break character has set the LBKDIF flag . IDLE is set only once even if the receive line remains idle for an extended period. 0b - No idle line detected. 1b - Idle line is detected.
19 OR	Receiver Overrun Flag OR is set when software fails to prevent the receive data register from overflowing with data. The OR bit is set immediately after the stop bit has been completely received for the dataword that overflows the buffer and all the other error flags (FE, NF, and PF) are prevented from setting. The data in the shift register is lost, but the data already in the LPUART data registers is not affected. If LBKDE is enabled and a LIN Break is detected, the OR field asserts if LBKDIF is not cleared before the next data character is received. While the OR flag is set, no additional data is stored in the receive FIFO even if sufficient room exists. To clear OR, write logic 1 to the OR flag. 0b - No overrun. 1b - Receive overrun (new LPUART data lost).
18 NF	Noise Flag The advanced sampling technique used in the receiver takes three samples in each of the received bits. If any of these samples disagrees with the rest of the samples within any bit time in the frame then noise is detected for that character. NF is set whenever the next character to be read from the DATA register is received with noise detected within the character. To clear NF, write logic 1 to the NF field. 0b - No noise detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Noise detected in the received character in the DATA register.
17 FE	Framing Error Flag FE is set whenever the next character to be read from the DATA register is received with logic 0 detected where a stop bit was expected. To clear FE, write logic 1 to the FE field. 0b - No framing error detected. This does not guarantee the framing is correct. 1b - Framing error.
16 PF	Parity Error Flag PF is set whenever the next character to be read from the DATA register is received when parity is enabled (PE = 1) and the parity bit in the received character does not agree with the expected parity value. To clear PF, write a logic 1 to the PF field. 0b - No parity error. 1b - Parity error.
15 MA1F	Match 1 Flag MA1F is set whenever the next character to be read from the DATA register matches MA1. To clear MA1F, write a logic 1 to the MA1F field. 0b - Received data is not equal to MA1 1b - Received data is equal to MA1
14 MA2F	Match 2 Flag MA2F is set whenever the next character to be read from the DATA register matches MA2. To clear MA2F, write a logic 1 to the MA2F field. 0b - Received data is not equal to MA2 1b - Received data is equal to MA2
13-10 —	Reserved
9-8 —	Reserved
7-2 —	Reserved
1 AME	Address Mark Enable Configures the location of the address mark when configured for MSB first transfers. This bit has no effect when configured for LSB first. This bit should be changed only when the transmitter and receiver are both disabled. 0b - Address mark in character is MSB.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Address mark in character is last bit before stop bit (or parity bit when enabled) and stored in DATA register at MSB (or MSB-1 when parity bit enabled).
0 LBKFE	LIN Break Flag Enable Enables the LIN Break Flag to assert whenever a LIN break character is detected. Unlike LBKDE, this does not impact data being stored in the receive data buffer, but does cause LBKDIF to assert whenever a LIN break is detected. Since a LIN break is longer than a normal character, the LIN break will trigger a write to the receive data register with the data bits clear and the Framing Error set. The character following the LIN break has the LINBRK flag set to indicate the previous character was a LIN break. This bit should be changed only when the transmitter and receiver are both disabled. 0b - LIN break detect is disabled. 1b - LIN break detect is enabled. LIN break character is detected at length of 11 bit times (if M = 0) or 12 (if M = 1) or 13 (M10 = 1).

73.6.1.8 LPUART Control Register (CTRL)

Offset

Register	Offset
CTRL	18h

Function

This read/write register controls various optional features of the LPUART system. This register should only be altered when the transmitter and receiver are both disabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R8T9	R9T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MA1IE	MA2IE	0		M7	IDLECFG			LOOPS	DOZEEN	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 R8T9	Receive Bit 8 / Transmit Bit 9 R8 is the ninth data bit received when the LPUART is configured for 9-bit or 10-bit data formats. When reading 9-bit or 10-bit data, read R8 before reading the DATA register. T9 is the tenth data bit transmitted when the LPUART is configured for 10-bit data formats. When writing 10-bit data, write T9 before writing the DATA register. If T9 does not need to change from its previous value, such as when it is used to generate address mark or parity, then it need not be written each time the DATA register is written. **NOTE** R8 is a read-only bit and T9 is a write-only bit, the value read is different from the value written.
30 R9T8	Receive Bit 9 / Transmit Bit 8 R9 is the tenth data bit received when the LPUART is configured for 10-bit data formats. When reading 10-bit data, read R9 before reading the DATA register T8 is the ninth data bit transmitted when the LPUART is configured for 9-bit or 10-bit data formats. When writing 9-bit or 10-bit data, write T8 before writing the DATA register. If T8 does not need to change from its previous value, such as when it is used to generate address mark or parity, then it need not be written each time the DATA register is written. **NOTE** R9 is a read-only bit and T8 is a write-only bit, the value read is different from the value written.
29 TXDIR	TXD Pin Direction in Single-Wire Mode When the LPUART is configured for single-wire half-duplex operation (LOOPS = RSRC = 1), this bit determines the direction of data at the TXD pin. When clearing TXDIR, the transmitter finishes receiving the current character (if any) before the receiver starts receiving data from the TXD pin. 0b - TXD pin is an input in single-wire mode. 1b - TXD pin is an output in single-wire mode.
28 TXINV	Transmit Data Inversion Setting this bit reverses the polarity of the transmitted data output. **NOTE** Setting TXINV inverts the TXD output for all cases: data bits, start and stop bits, break, and idle. 0b - Transmit data not inverted. 1b - Transmit data inverted.
27 ORIE	Overrun Interrupt Enable This bit enables the overrun flag (OR) to generate hardware interrupt requests.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - OR interrupts disabled; use polling. 1b - Hardware interrupt is requested when OR is set.
26 NEIE	Noise Error Interrupt Enable This bit enables the noise flag (NF) to generate hardware interrupt requests. 0b - NF interrupts disabled; use polling. 1b - Hardware interrupt is requested when NF is set.
25 FEIE	Framing Error Interrupt Enable This bit enables the framing error flag (FE) to generate hardware interrupt requests. 0b - FE interrupts disabled; use polling. 1b - Hardware interrupt is requested when FE is set.
24 PEIE	Parity Error Interrupt Enable This bit enables the parity error flag (PF) to generate hardware interrupt requests. 0b - PF interrupts disabled; use polling). 1b - Hardware interrupt is requested when PF is set.
23 TIE	Transmit Interrupt Enable Enables STAT[TDRE] to generate interrupt requests. 0b - Hardware interrupts from TDRE disabled. 1b - Hardware interrupt is requested when TDRE flag is 1.
22 TCIE	Transmission Complete Interrupt Enable for TCIE enables the transmission complete flag, TC, to generate interrupt requests. 0b - Hardware interrupts from TC disabled. 1b - Hardware interrupt is requested when TC flag is 1.
21 RIE	Receiver Interrupt Enable Enables STAT[RDRF] to generate interrupt requests. 0b - Hardware interrupts from RDRF disabled. 1b - Hardware interrupt is requested when RDRF flag is 1.
20 ILIE	Idle Line Interrupt Enable ILIE enables the idle line flag, STAT[IDLE], to generate interrupt requests. 0b - Hardware interrupts from IDLE disabled; use polling. 1b - Hardware interrupt is requested when IDLE flag is 1.
19	Transmitter Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TE	Enables the LPUART transmitter. TE can also be used to queue an idle preamble by clearing and then setting TE. When TE is cleared, this register bit reads as 1 until the transmitter has completed the current character and the TXD pin is tristated. A single idle character can also be queued by writing to the transmit FIFO with DATA[FRETSC] set and DATA[R9T9] set. 0b - Transmitter disabled. 1b - Transmitter enabled.
18 RE	Receiver Enable Enables the LPUART receiver. When RE is written to 0, this register bit reads as 1 until the receiver finishes receiving the current character (if any). 0b - Receiver disabled. 1b - Receiver enabled.
17 RWU	Receiver Wakeup Control This field can be set to place the LPUART receiver in a standby state. RWU automatically clears when an RWU event occurs, that is, an IDLE event when CTRL[WAKE] is clear or an address match when CTRL[WAKE] is set and STAT[RWUID] is clear. **NOTE** RWU must be set only with CTRL[WAKE] = 0 (wakeup on idle) if the channel is currently not idle. This can be determined by STAT[RAF]. If the flag is set to wake up an IDLE event and the channel is already idle, it is possible that the LPUART will discard data. This is because the data must be received or a LIN break detected after an IDLE is detected before IDLE is allowed to be reasserted. 0b - Normal receiver operation. 1b - LPUART receiver in standby waiting for wakeup condition.
16 SBK	Send Break Writing a 1 and then a 0 to SBK queues a break character in the transmit data stream. Additional break characters of 9 to 13 bits, or 12 to 15 bits if STAT[BRK13] is set, bit times of logic 0 are queued as long as SBK is set. Depending on the timing of the set and clear of SBK relative to the character currently being transmitted, a second break character may be queued before software clears SBK. If software takes too long to clear SBK (for example, it is not cleared by the end of the 1st break character), then a 2nd break character is sent. This is compared to queuing a break character through the transmit FIFO which guarantees only one is sent. A single break character can also be queued by writing to the transmit FIFO with DATA[FRETSC] set and DATA[R9T9] clear. 0b - Normal transmitter operation. 1b - Queue break character(s) to be sent.
15	Match 1 Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
MA1IE	0b - MA1F interrupt disabled 1b - MA1F interrupt enabled
14 MA2IE	Match 2 Interrupt Enable 0b - MA2F interrupt disabled 1b - MA2F interrupt enabled
13-12 —	Reserved
11 M7	7-Bit Mode Select This bit should be changed only when the transmitter and receiver are both disabled. 0b - Receiver and transmitter use 8-bit to 10-bit data characters. 1b - Receiver and transmitter use 7-bit data characters.
10-8 IDLECFG	Idle Configuration Configures the number of idle characters that must be received before the IDLE flag is set. 000b - 1 idle character 001b - 2 idle characters 010b - 4 idle characters 011b - 8 idle characters 100b - 16 idle characters 101b - 32 idle characters 110b - 64 idle characters 111b - 128 idle characters
7 LOOPS	Loop Mode Select When LOOPS is set, the RXD pin is disconnected from the LPUART and the transmitter output is internally connected to the receiver input. The transmitter and the receiver must be enabled to use the loop function. 0b - Normal operation - RXD and TXD use separate pins. 1b - Loop mode or single-wire mode where transmitter outputs are internally connected to receiver input (see RSRC bit).
6 DOZEEN	Doze Enable 0b - LPUART is enabled in Doze mode. 1b - LPUART is disabled in Doze mode , but remains active when not in Doze mode .
5	Receiver Source Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
RSRC	This field has no meaning or effect unless the LOOPS field is set. When LOOPS is set, the RSRC field determines the source for the receiver shift register input. 0b - Provided LOOPS is set, RSRC is cleared, selects internal loop back mode and the LPUART does not use the RXD pin. 1b - Single-wire LPUART mode where the TXD pin is connected to the transmitter output and receiver input.
4 M	9-Bit or 8-Bit Mode Select 0b - Receiver and transmitter use 8-bit data characters. 1b - Receiver and transmitter use 9-bit data characters.
3 WAKE	Receiver Wakeup Method Select Determines which condition wakes the LPUART when RWU=1 and MATCFG=00 (WAKE must be set when MATCFG=11): • Address mark in the bit preceding the stop bit (or bit preceding the parity bit when parity is enabled) of the received data character. • An idle condition on the receive pin input signal. 0b - Configures RWU for idle-line wakeup. 1b - Configures RWU with address-mark wakeup.
2 ILT	Idle Line Type Select Determines when the receiver starts counting logic 1s as idle character bits. The count begins either after a valid start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit can cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. **NOTE** In case the LPUART is programmed with ILT = 1, a logic 0 is automatically shifted after a received stop bit, therefore resetting the idle count. 0b - Idle character bit count starts after start bit. 1b - Idle character bit count starts after stop bit.
1 PE	Parity Enable Enables hardware parity generation and checking. When parity is enabled, the bit immediately before the stop bit is treated as the parity bit. 0b - No hardware parity generation or checking. 1b - Parity enabled.
0 PT	Parity Type Provided parity is enabled (PE = 1), this bit selects even or odd parity. Odd parity means the total number of logic 1 bits in the data character, including the parity bit, is odd. Even parity means the total number of 1s in the data character, including the parity bit, is even.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Even parity. 1b - Odd parity.

73.6.1.9 LPUART Data Register (DATA)

Offset

Register	Offset
DATA	1Ch

Function

NOTE

This register is two separate registers. Reads return the contents of the read-only receive FIFO and writes go to the write-only transmit FIFO.

Reads and writes of this register are also involved in the automatic flag clearing mechanisms for some of the LPUART status flags. This register can be written using 8-bit, 16-bit or 32-bit writes. An 8-bit write to DATA[7:0] will push { R8T9, R9T8, DATA[7:0] } the transmit FIFO with TSC clear. A 16-bit or 32-bit write will push the data written into the FIFO and does not update the value in R8T9 or R9T8.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NOISY	PARITYE	FRETSC	RXEMPT	IDLINE	LINBRK	R9T9	R8T8	R7T7	R6T6	R5T5	R4T4	R3T3	R2T2	R1T1	R0T0
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15	Noisy Data Received

Table continues on the next page...

Table continued from the previous page...

Field	Function
NOISY	The current received dataword contained in DATA[R9:R0] is received with noise. 0b - The dataword is received without noise. 1b - The data is received with noise.
14 PARITYE	Parity Error The current received dataword contained in DATA[R9:R0] is received with a parity error. 0b - The dataword is received without a parity error. 1b - The dataword is received with a parity error.
13 FRETSC	Frame Error / Transmit Special Character For reads, indicates the current received dataword contained in DATA[R9:R0] was received with a frame error. For writes, indicates a break or idle character is to be transmitted instead of the contents in DATA[T9:T0]. T9 is used to indicate a break character when 0 and a idle character when 1, the contents of DATA[T8:T0] should be zero. 0b - The dataword is received without a frame error on read, or transmit a normal character on write. 1b - The dataword is received with a frame error, or transmit an idle or break character on transmit.
12 RXEMPT	Receive Buffer Empty Asserts when there is no data in the receive buffer. This field does not take into account data that is in the receive shift register. 0b - Receive buffer contains valid data. 1b - Receive buffer is empty, data returned on read is not valid.
11 IDLINE	Idle Line Indicates the receiver line was idle before receiving the character in DATA[9:0]. Unlike the IDLE flag, this bit can set for the first character received when the receiver is first enabled. 0b - Receiver was not idle before receiving this character. 1b - Receiver was idle before receiving this character.
10 LINBRK	LIN Break Indicates the receiver line detected a LIN break before receiving the character in DATA[9:0]. This flag requires LBKDF to be set. 0b - Receiver did not detect LIN break before this character, or LIN break detect circuitry disabled. 1b - Receiver detected a LIN break before receiving this character.
9 R9T9	R9T9 Read receive FIFO bit 9 or write transmit FIFO bit 9.
8	R8T8

Table continues on the next page...

Table continued from the previous page...

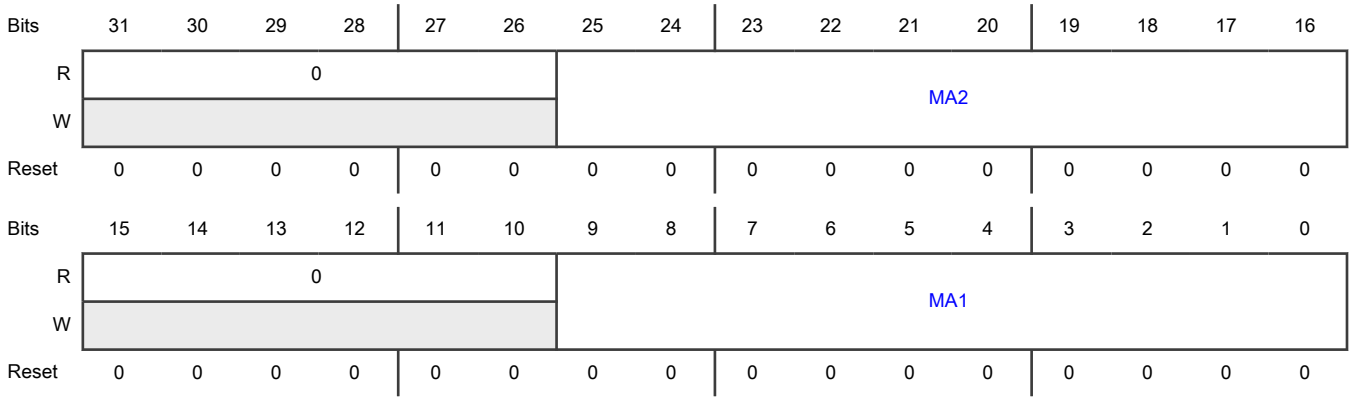
Field	Function
R8T8	Read receive FIFO bit 8 or write transmit FIFO bit 8.
7 R7T7	R7T7 Read receive FIFO bit 7 or write transmit FIFO bit 7.
6 R6T6	R6T6 Read receive FIFO bit 6 or write transmit FIFO bit 6.
5 R5T5	R5T5 Read receive FIFO bit 5 or write transmit FIFO bit 5.
4 R4T4	R4T4 Read receive FIFO bit 4 or write transmit FIFO bit 4.
3 R3T3	R3T3 Read receive FIFO bit 3 or write transmit FIFO bit 3.
2 R2T2	R2T2 Read receive FIFO bit 2 or write transmit FIFO bit 2.
1 R1T1	R1T1 Read receive FIFO bit 1 or write transmit FIFO bit 1.
0 R0T0	R0T0 Read receive FIFO bit 0 or write transmit FIFO bit 0.

73.6.1.10 LPUART Match Address Register (MATCH)

Offset

Register	Offset
MATCH	20h

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 MA2	Match Address 2 The MA1 and MA2 fields are compared to input data addresses when the most significant bit is set and the associated BAUD[MAEN] bit is set. If a match occurs, the following data is transferred to the DATA register. If a match fails, the following data is discarded. Software should only write a MAx field when the associated BAUD[MAEN] bit is clear.
15-10 —	Reserved
9-0 MA1	Match Address 1 The MA1 and MA2 fields are compared to input data addresses when the most significant bit is set and the associated BAUD[MAEN] bit is set. If a match occurs, the following data is transferred to the DATA register. If a match fails, the following data is discarded. Software should only write a MAx field when the associated BAUD[MAEN] bit is clear.

73.6.1.11 LPUART Modem IrDA Register (MODIR)

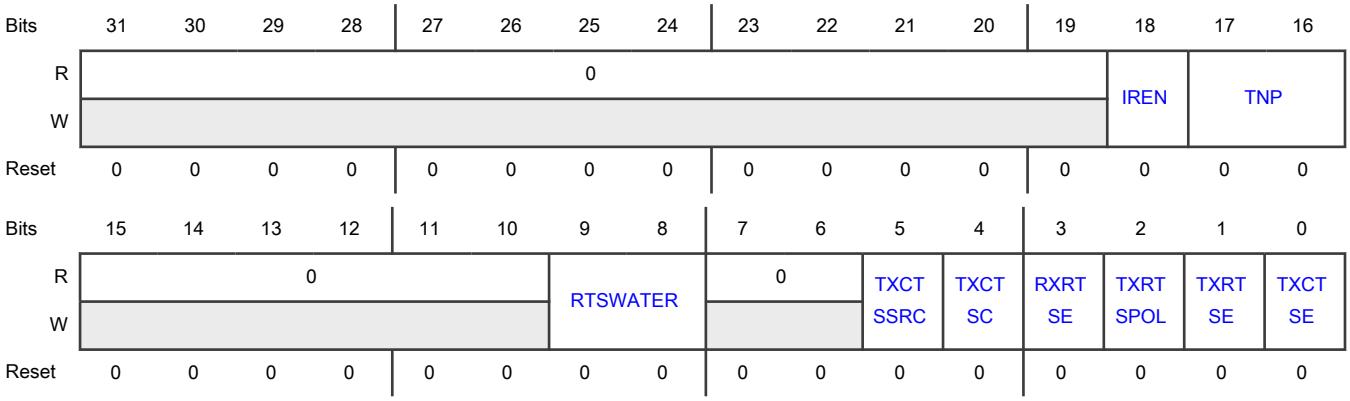
Offset

Register	Offset
MODIR	24h

Function

The MODIR register controls options for setting the modem configuration.

Diagram



Fields

Field	Function
31-19 —	Reserved
18 IREN	Infrared enable Enables/disables the infrared modulation/demodulation. This bit should be changed only when the transmitter and receiver are both disabled. 0b - IR disabled. 1b - IR enabled.
17-16 TNP	Transmitter narrow pulse Configures whether the LPUART transmits a 1/OSR, 2/OSR, 3/OSR or 4/OSR narrow pulse when IR is enabled. This bit should be changed only when the transmitter and receiver are both disabled. The IR pulse width should be configured to less than half of the oversampling ratio. Common pulse widths are 3/16, 1/16, 1/32 or 1/4 of the bit length. These can be configured by selecting the appropriate oversample ratio and pulse width. 00b - 1/OSR. 01b - 2/OSR. 10b - 3/OSR. 11b - 4/OSR.
15-10 —	Reserved
9-8 RTSWATER	Receive RTS Configuration Configures the assertion and negation of the RX RTS_B output. The RX RTS_B output negates when the number of empty words in the receive FIFO is greater or equal to RTSWATER. If RTSWATER is configured to zero, RTS_B negates when the receive FIFO is full. For the purpose of RX RTS_B generation, the number of words in the receive FIFO updates when a start bit is detected. This supports additional latency between

Table continues on the next page...

Table continued from the previous page...

Field	Function
	RTS_B negation and the external transmitter ceasing transmission. If both RX RTS_B and address/data matching is enabled, then RTS_B could assert at the end of a character if there is not a match. This field should be changed only when the receiver is disabled.
7-6 —	Reserved
5 TXCTSSRC	Transmit CTS Source Configures the source of the CTS input. 0b - CTS input is the CTS_B pin. 1b - CTS input is an internal connection to the receiver address match result.
4 TXCTSC	Transmit CTS Configuration Configures if the CTS state is checked at the start of each character or only when the transmitter is idle. 0b - CTS input is sampled at the start of each character. 1b - CTS input is sampled when the transmitter is idle.
3 RXRTSE	Receiver request-to-send enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun. This bit should be changed only when the receiver is disabled. **NOTE** Do not set both RXRTSE and TXRTSE. 0b - The receiver has no effect on RTS. 1b - RTS is deasserted if the receiver data register is full or a start bit has been detected that would cause the receiver data register to become full. RTS is asserted if the receiver data register is not full and has not detected a start bit that would cause the receiver data register to become full.
2 TXRTSPOL	Transmitter request-to-send polarity Controls the polarity of the transmitter RTS. TXRTSPOL does not affect the polarity of the receiver RTS. RTS remains negated in the active low state unless TXRTSE is set. This bit should be changed only when the transmitter is disabled. 0b - Transmitter RTS is active low. 1b - Transmitter RTS is active high.
1 TXRTSE	Transmitter request-to-send enable Controls RTS before and after a transmission. This bit should be changed only when the transmitter is disabled. 0b - The transmitter has no effect on RTS.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - When a character is placed into an empty transmit shift register, RTS asserts one bit time before the start bit is transmitted. RTS deasserts one bit time after all characters in the transmitter FIFO and shift register are completely sent, including the last stop bit.
0 TXCTSE	Transmitter clear-to-send enable TXCTSE controls the operation of the transmitter. TXCTSE can be set independently from the state of TXRTSE and RXRTSE. 0b - CTS has no effect on the transmitter. 1b - Enables clear-to-send operation. The transmitter checks the state of CTS each time it is ready to send a character. If CTS is asserted, the character is sent. If CTS is deasserted, the signal TXD remains in the mark state and transmission is delayed until CTS is asserted. Changes in CTS as a character is being sent do not affect its transmission.

73.6.1.12 LPUART FIFO Register (FIFO)

Offset

Register	Offset
FIFO	28h

Function

This register provides the ability for the programmer to turn on and off FIFO functionality. It also provides the size of the FIFO that has been implemented. This register may be read at any time. This register must be written only when CTRL[RE] and CTRL[TE] are cleared/not set and when the FIFO is empty.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								TXEMPT	RXEMPT	0				TXOF	RXUF
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	RXIDEN			TXOFE	RXUFE	TXFE	TXFIFOSIZE			RXFE	RXFIFOSIZE		
W	TXFLUSH	RXFLUSH														
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

Fields

Field	Function
31-24 —	Reserved
23 TXEMPT	Transmit FIFO/Buffer Empty Asserts when there is no data in the Transmit FIFO/Buffer. This field does not take into account data that is in the transmit shift register. 0b - Transmit buffer is not empty. 1b - Transmit buffer is empty.
22 RXEMPT	Receive FIFO/Buffer Empty Asserts when there is no data in the Receive FIFO/Buffer. This field does not take into account data that is in the receive shift register. 0b - Receive buffer is not empty. 1b - Receive buffer is empty.
21-18 —	Reserved
17 TXOF	Transmitter FIFO Overflow Flag Indicates that more data has been written to the transmit FIFO than it can hold. This field asserts regardless of the value of TXOFE. However, an interrupt is issued to the host only if TXOFE is set. Write 1 to clear this flag. 0b - No transmit FIFO overflow has occurred since the last time the flag was cleared. 1b - At least one transmit FIFO overflow has occurred since the last time the flag was cleared.
16 RXUF	Receiver FIFO Underflow Flag Indicates that more data has been read from the receive FIFO than was present. This field asserts regardless of the value of RXUFE. However, an interrupt is issued to the host only if RXUFE is set. Write a logic 1 to clear RXUF. 0b - No receive FIFO underflow has occurred since the last time the flag was cleared. 1b - At least one receive FIFO underflow has occurred since the last time the flag was cleared.
15 TXFLUSH	Transmit FIFO Flush Writing to this field causes all data that is stored in the transmit FIFO to be flushed. This does not affect data that is in the transmit shift register. 0b - No flush operation occurs. 1b - All data in the transmit FIFO is cleared out.
14 RXFLUSH	Receive FIFO Flush Writing to this field causes all data that is stored in the receive FIFO to be flushed. This does not affect data that is in the receive shift register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No flush operation occurs. 1b - All data in the receive FIFO/buffer is cleared out.
13 —	Reserved
12-10 RXIDEN	Receiver Idle Empty Enable When set, enables the assertion of RDRF when the receiver is idle for a number of idle characters and the FIFO is not empty. 000b - Disable RDRF assertion due to partially filled FIFO when receiver is idle. 001b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 1 character. 010b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 2 characters. 011b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 4 characters. 100b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 8 characters. 101b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 16 characters. 110b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 32 characters. 111b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 64 characters.
9 TXOFE	Transmit FIFO Overflow Interrupt Enable When this field is set, the TXOF flag generates an interrupt to the host. 0b - TXOF flag does not generate an interrupt to the host. 1b - TXOF flag generates an interrupt to the host.
8 RXUFE	Receive FIFO Underflow Interrupt Enable When this field is set, the RXUF flag generates an interrupt to the host. 0b - RXUF flag does not generate an interrupt to the host. 1b - RXUF flag generates an interrupt to the host.
7 TXFE	Transmit FIFO Enable When this field is set, the built-in FIFO structure for the transmit buffer is enabled. The size of the FIFO structure is indicated by TXFIFOSIZE. If this field is not set, the transmit buffer operates as a FIFO of depth one dataword regardless of the value in TXFIFOSIZE. Both CTRL[TE] and CTRL[RE] must be cleared prior to changing this field. 0b - Transmit FIFO is not enabled. Buffer depth is 1. 1b - Transmit FIFO is enabled. Buffer depth is indicated by TXFIFOSIZE.
6-4 TXFIFOSIZE	Transmit FIFO Buffer Depth The maximum number of transmit datawords that can be stored in the transmit buffer. This field is read-only.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - Transmit FIFO/Buffer depth = 1 dataword. 001b - Transmit FIFO/Buffer depth = 4 datawords. 010b - Transmit FIFO/Buffer depth = 8 datawords. 011b - Transmit FIFO/Buffer depth = 16 datawords. 100b - Transmit FIFO/Buffer depth = 32 datawords. 101b - Transmit FIFO/Buffer depth = 64 datawords. 110b - Transmit FIFO/Buffer depth = 128 datawords. 111b - Transmit FIFO/Buffer depth = 256 datawords
3 RXFE	Receive FIFO Enable When this field is set, the built-in FIFO structure for the receive buffer is enabled. The size of the FIFO structure is indicated by the RXFIFOSIZE field. If this field is not set, the receive buffer operates as a FIFO of depth one dataword regardless of the value in RXFIFOSIZE. Both CTRL[TE] and CTRL[RE] must be cleared prior to changing this field. 0b - Receive FIFO is not enabled. Buffer depth is 1. 1b - Receive FIFO is enabled. Buffer depth is indicted by RXFIFOSIZE.
2-0 RXFIFOSIZE	Receive FIFO Buffer Depth The maximum number of receive datawords that can be stored in the receive buffer before an overrun occurs. This field is read-only. 000b - Receive FIFO/Buffer depth = 1 dataword. 001b - Receive FIFO/Buffer depth = 4 datawords. 010b - Receive FIFO/Buffer depth = 8 datawords. 011b - Receive FIFO/Buffer depth = 16 datawords. 100b - Receive FIFO/Buffer depth = 32 datawords. 101b - Receive FIFO/Buffer depth = 64 datawords. 110b - Receive FIFO/Buffer depth = 128 datawords. 111b - Receive FIFO/Buffer depth = 256 datawords.

73.6.1.13 LPUART Watermark Register (WATER)

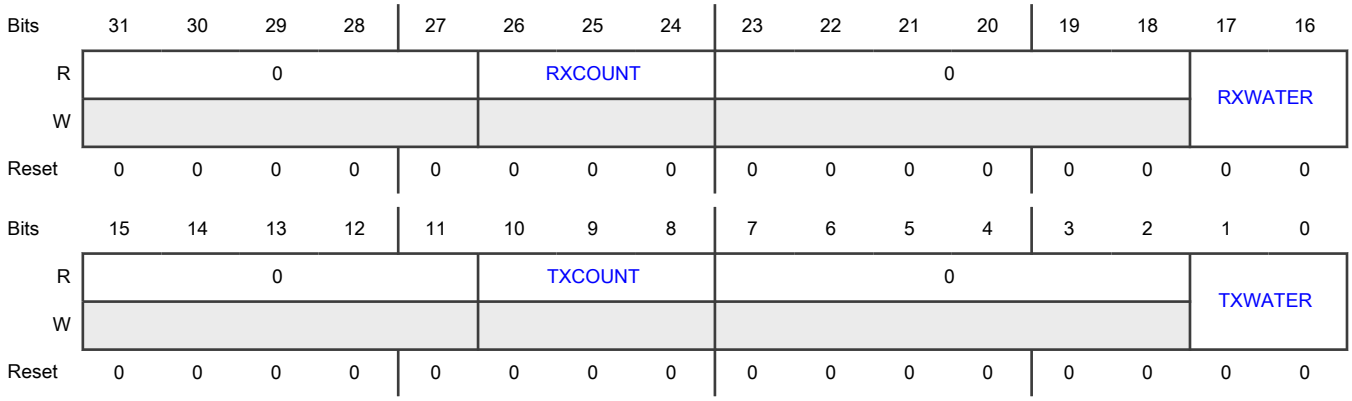
Offset

Register	Offset
WATER	2Ch

Function

This register provides the ability to set a programmable threshold for notification of needing additional transmit data. This register may be read at any time but must be written only when CTRL[TE] is not set.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 RXCOUNT	Receive Counter The value in this register indicates the number of datawords that are in the receive FIFO/buffer. If a dataword is being received, that is, in the receive shift register, it is not included in the count. This value may be used in conjunction with FIFO[RXFIFOSIZE] to calculate how much room is left in the receive FIFO/buffer.
23-18 —	Reserved
17-16 RXWATER	Receive Watermark When the number of datawords in the receive FIFO/buffer is greater than the value in this register field, an interrupt or a DMA request is generated. For proper operation, the value in RXWATER must be set to be less than the receive FIFO/buffer size as indicated by FIFO[RXFIFOSIZE] and FIFO[RXFE].
15-11 —	Reserved
10-8 TXCOUNT	Transmit Counter The value in this register indicates the number of datawords that are in the transmit FIFO/buffer. If a dataword is being transmitted, that is, in the transmit shift register, it is not included in the count. This value may be used in conjunction with FIFO[TXFIFOSIZE] to calculate how much room is left in the transmit FIFO/buffer.
7-2 —	Reserved
1-0 TXWATER	Transmit Watermark

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When the number of datawords in the transmit FIFO/buffer is equal to or less than the value in this register field, an interrupt or a DMA request is generated. For proper operation, the value in TXWATER must be set to be less than the size of the transmit buffer/FIFO size as indicated by FIFO[TXFIFOSIZE] and FIFO[TXFE].

73.6.1.14 Data read-only Register (DATARO)

Offset

Register	Offset
DATARO	30h

Function

Returns the first entry in the receive FIFO, but does not pull the data from the FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA															
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Receive Data Returns first entry from the receive FIFO, the bit definition is the same as the DATA register.

73.7 Glossary

Baud rate Number of bits transmitted or received per second by the LPUART

Break character Break character is generated when the transmitter is holding the data line at the space level for at least 1 character time

Oversampling	Number of times the receive circuitry samples the receive input per baud period (i.e. per data bit)

Chapter 74
Quad Serial Peripheral Interface (QuadSPI)

74.1 Chip-specific QuadSPI information

74.1.1 QuadSPI configuration

Table 540. QuadSPI instances

Instance	MWCT2D16S/MWCT2D17S	MWCT2015S/MWCT2016S
QuadSPI	Yes	No

Table 541. QuadSPI configuration details

Configuration	MWCT2x16S/MWCT2D17S
QuadSPI Tx FIFO size	32 words
QuadSPI Rx FIFO size	32 words
Look Up Table Size	4 words
AHB Buffer Size	256 Bytes

Table 542. Supported data rates at different frequencies

Supported data rates	80 MHz	120 MHz
4 pin SDR	Yes	Yes
4 pin DDR	No	No

NOTE

Boot from QuadSPI is not supported but execution from external memory is supported.

Table 543. Features supported

Feature	MWCT2x16S/MWCT2D17S
AHB Write	No
Data learning feature	No
OTFAD (On-the-fly-AES-decryption engine)	No
DDR mode	No
HyperRAM	No
HyperFlash	No
External DQS (Data Strobe)	No
Boot from QuadSPI interface	No

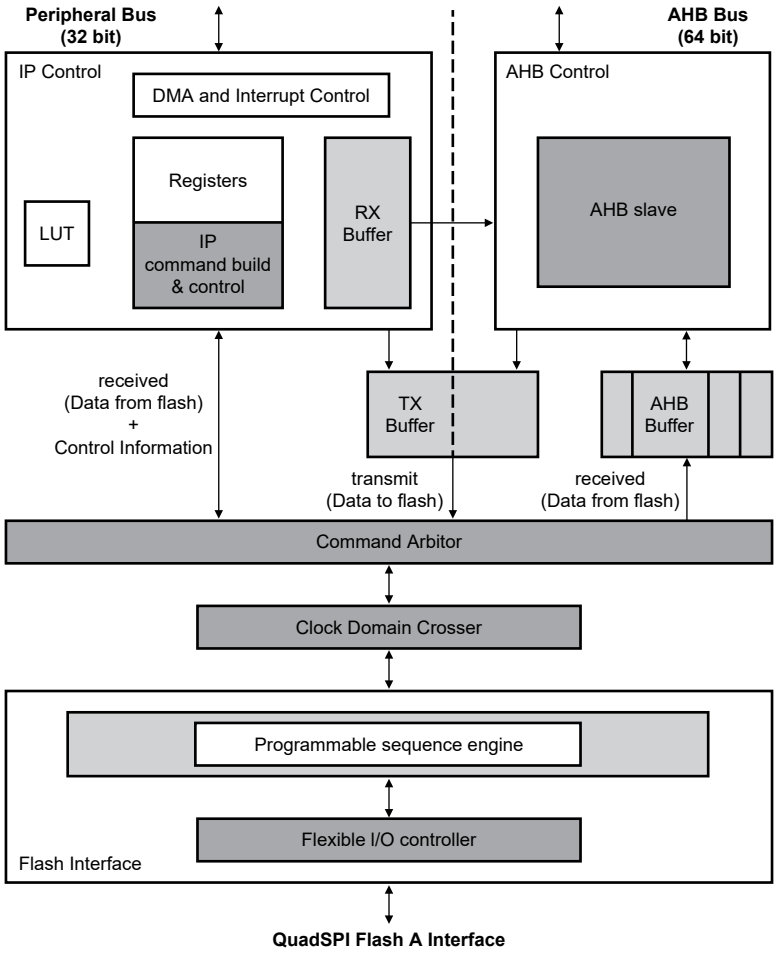


Figure 430. Block diagram

74.1.2 Supported read modes

The table below provides an overview of the QuadSPI read modes.

Table 544. QuadSPI read modes

Read modes		SDR support (QuadSPI_MCR [DDR_EN]=0)	QuadSPI_MCR [DQS_EN]	QuadSPI_MCR [DQS_FA_SEL]	DLL/Data learning support
DQS sampling method	Pad loopback	Yes	1	01	No

74.1.3 QuadSPI initialization sequence

Following initialization sequence should be followed for proper QuadSPI operation:

- Enable QuadSPI module by MC_ME peripheral clock enable (register PRTNx_COFBy_CLKEN present within MC_ME). Refer MC_ME chapter for peripheral mapping.
- Configure the SIUL2 registers MSCR[OBE] as 1 and MSCR[SSS] as 0 for QuadSPI_SCKFA pin.
- Initialize QuadSPI SCKFA by writing a sequence of 1010 to the SIUL2 register GPDO[PDO_a] for QuadSPI_SCKFA pin.
- Configure the SIUL2 register MSCR[OBE] back as 0 for the pins.

- Initiate a dummy flash read to reset all DQS flops by itself and any crossing from DQS domain to IPS/AHB is taken care by CDC logic.
- Initiate a peripheral software reset to QuadSPI controller by writing to QuadSPI controller's MCR.
- Post this initialization sequence, the QuadSPI will work in intended deterministic manner.

NOTE

QuadSPI initialization is to be done before using QuadSPI after each functional reset.

74.1.4 Pad clock loopback

This chip supports pad clock loopback. The QuadSPI can be configured to use clock loopback to sample input data. SCK is delayed by the SCK pin output delay, plus the SCK pin input delay using pad loopback, and is configured by setting QuadSPI config registers SOCCR[SOCCFG] and MCR[DQS_FA_SEL]. Enabling the loopback version of SCK can improve the setup time of the input data from the Flash.

For details of these register, see QuadSPI register descriptions.

74.1.5 QuadSPI SOC Configuration register SOCCR[SOCCFG] implementation

The QuadSPI SOC Configuration register QuadSPI_SOCCR[SOCCFG] register is used to control dummy loopback pads and obe_pull_timing_relax_b. Below is the description of it's bits:

Table 545. SOCCR[SOCCFG] implementation

Bit	Description
Bit[0]	obe_pull_timing_relax_b : enables the timing relaxation by pulling obe for pad 1 for half cycle, if 0 then enabled else disabled.
Bit[1]	ibe of QSPIA_SCK_DUMMY pad. 0: Disable input reciever 1: Enable input reciever.
Bit[2]	obe of QSPIA_SCK_DUMMY pad. 0: Disable output driver 1: Enable output driver.
Bit[3]	dse of QSPIA_SCK_DUMMY pad. 0: Disable drive strength 1: Enable drive strength.
Bit[4]	pue of QSPIA_SCK_DUMMY pad. 0: Disable internal pullup or pulldown resistor 1: Enable internal pullup or pulldown resistor.
Bit[5]	pus of QSPIA_SCK_DUMMYpad. 0: Enable internal pulldown resistor if pue is set 1: Enable internal pullup resistor if pue is set.
Bit[6]	sre of QSPIA_SCK_DUMMY pad. 0: Disable slew rate control 1: Enable slew rate control.
Bit[31:7]	Reserved

To Enable the Quadspi dummy PAD loopback use following settings

For Flash-A: MCR[DQS_FB_SEL] = 0x2

SOCCR[SOCCFG] = 0x0000000E (ibe=1, obe=1, dse=1 and sre=0)

74.1.6 QuadSPI AHB Buffer write access control

In MWCT2D17S and MWCT2x16S devices, the QuadSPI AHB buffer does not support writes. A write to QuadSPI AHB buffer results in error response from QuadSPI.

QuadSPI might behave unexpectedly in case if its AHB buffer is written with QuadSPI MCR[MDIS] =1. In cases wherein QuadSPI is disabled with above MDIS control bit, XRDC must also be appropriately configured to block the accesses to QSPI AHB buffer and indicate an error response.

74.2 Introduction

The QuadSPI module acts as an interface to a single serial flash memory device, with up to four bidirectional data lines.

74.2.1 Features

QuadSPI supports the following features:

- Flexible sequence engine to support various flash memory vendor devices. As there is no specific standard, the module supports various kinds of flash memories from different vendors. See Serial flash memory devices for example sequences.
- Single, dual, and quad modes of operation supported for Quad flash memories
- AHB master to read RX buffer data through AMBA AHB (64-bit width interface) or IPS registers space (32-bit access) and fill TX buffer via IPS registers space (32-bit access)
  - AHB master can be a DMA with a configurable inner loop size
- Multi-master accesses are allowed
  - Flexible and configurable buffer for each master—total available buffer size is 256 bytes.
- All AHB accesses to flash memory devices are directly memory mapped to the chip system memory
- Programmable sequence engine to cater to future command/protocol changes and ability to support all existing vendor commands and operations. The software needs to select the corresponding sequence according to the connected flash memory device.
  - Support for 3-byte and 4-byte addressing

74.2.2 RX buffer push event

To add the valid entries into the RX buffer

By default, each buffer push event adds two entries to the RX buffer because the interface to the serial clock domain is 64 bits in width. Depending on the number of bytes read from the serial flash memory device, it is possible for the very last buffer push event that only one entry is added.

RBSR[RDBFL] is incremented by the number of entries added to the RX buffer.

74.2.3 RX buffer POP event

To remove valid entries from the RX buffer

Each buffer POP event removes (RBCT[WMRK] + 1) valid entries from the buffer. BSR[RDBFL] is decremented by the same number and RBSR[RDCTR] is incremented accordingly.

74.2.4 Block diagram

The following figure shows a block diagram of the QuadSPI module.

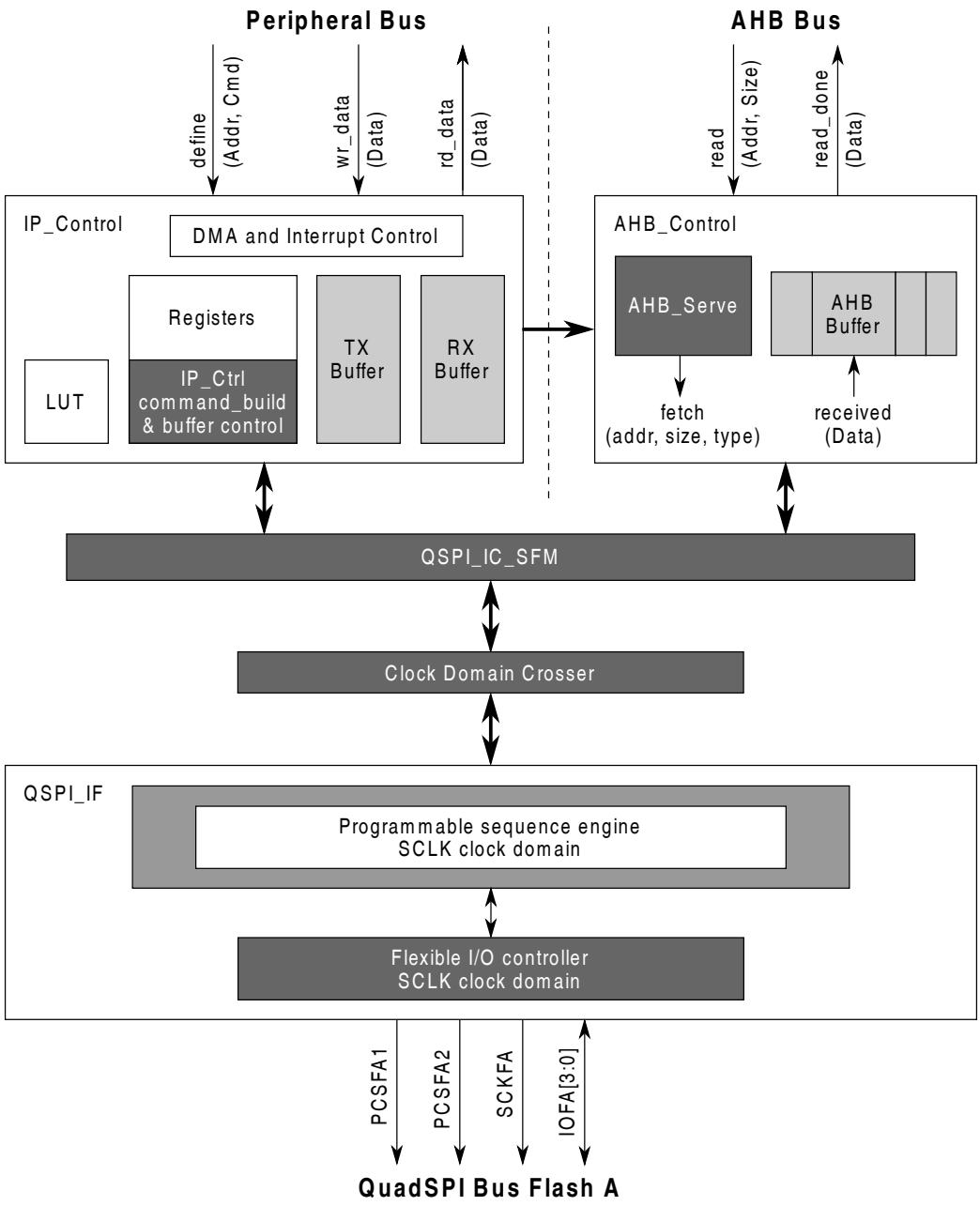


Figure 431. QuadSPI Block Diagram

74.2.5 QuadSPI modes of operation

QuadSPI supports the following modes of operation:

- Normal mode: You can use this mode for write or read accesses to an external serial flash memory device. See Normal mode for details.
  - Serial flash memory write: You can program data into the flash memory through the IP interface only. See Flash memory programming for details.
  - Serial flash memory read: Read the contents of the serial flash memory device. Two separate read channels are available through the RX buffer and AHB buffer. See Flash memory read for details.

- Module Disable mode: You can use this mode for disabling serial flash memory clock and AHB command. The clock to the non-memory mapped logic in QuadSPI can be stopped in the Module Disable mode. The module enters the mode by setting MCR[MDIS].

74.3 External signal description

This section provides the external signal information for the QuadSPI module.

The following table lists the external signals belonging to the module in conjunction with the different modes of operation.

Table 546. Signal properties

Signal name	Function	Direction	Description
PCSFA1	Peripheral Chip Select Flash Memory A1	O	This signal is the chip select for the serial flash memory device A1 that represents the first of the two flash memory devices that share IOFA.
PCSFA2	Peripheral Chip Select Flash Memory A2	O	This signal is the chip select for the serial flash memory device A2 that represents the the second of the two flash memory devices that share IOFA.
SCKFA	Serial Clock Flash Memory A	O	This signal is the serial clock output to the serial flash memory device A.
IOFA[3:0]	Serial I/O Flash memory A	I/O	These signals are the data I/O lines to/from the serial flash memory device A. See Driving external signals for details about the signal drive and timing behavior. Note that the signal pins of the serial flash memory device may change their function according to the SFM Command executed, leaving them as control inputs when single and dual instructions are executed. The module supports driving these inputs to dedicated values. In single I/O mode, QuadSPI drives data on IOFA[0] and expects data on IOFA[1].

74.3.1 Driving external signals

Single/dual/quad instructions

Depending on the serial flash memory device connected to the QuadSPI module, there are instructions using a different number of data lines:

- Single pad: Single line I/O with one data out and one data in line to/from the serial flash memory device
- Dual pad: Dual line I/O with two bidirectional I/O lines, driven alternatively by the serial flash memory device or the QuadSPI module
- Quad pad: Quad line I/O with four bidirectional I/O lines, driven alternatively by the serial flash memory device or the QuadSPImodule

The different phases of the serial flash memory access scheme are shown in the following figure.

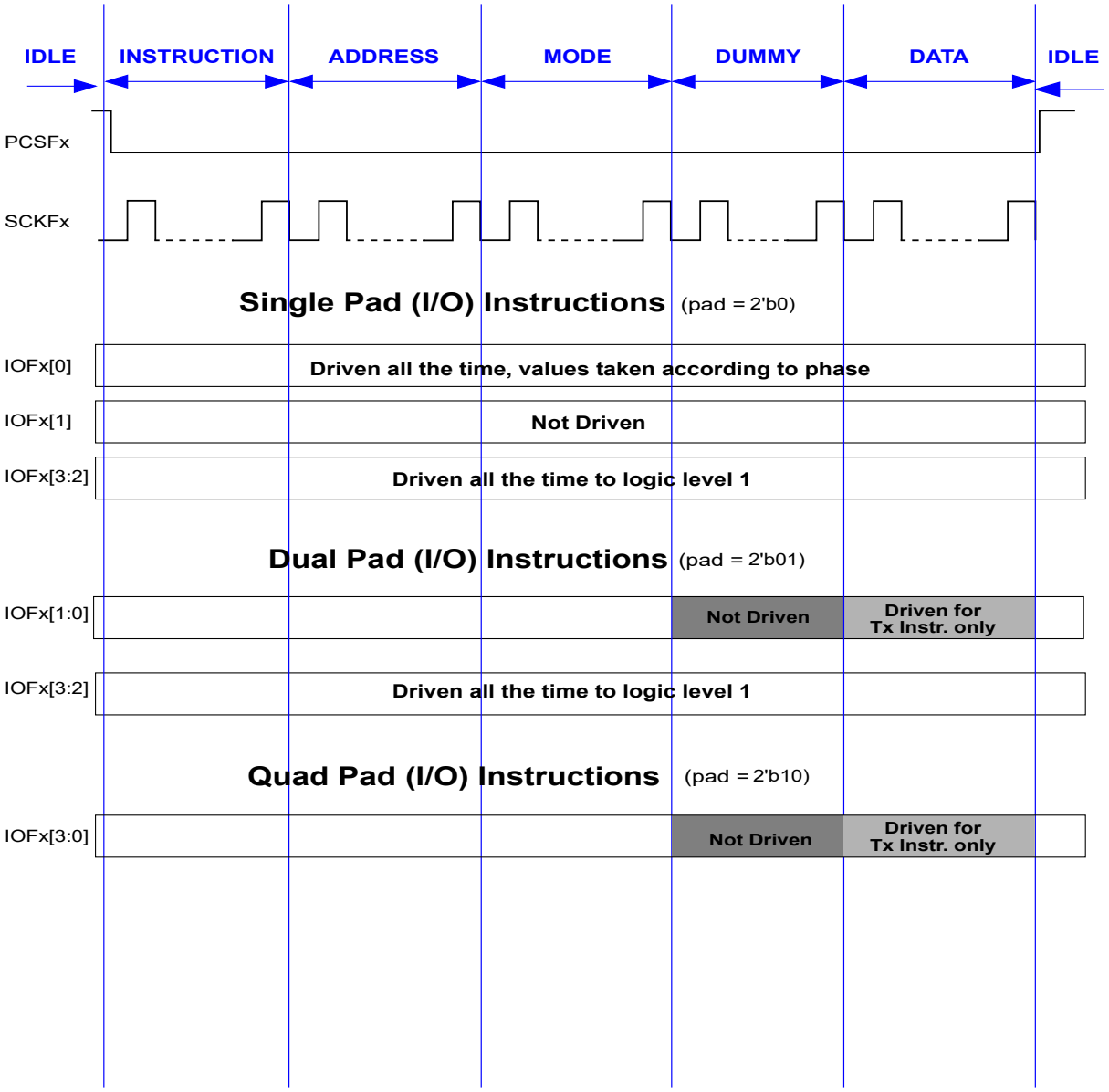


Figure 432. Serial flash memory access scheme

Following are the different phases and the I/O driving characteristics of the QuadSPI module:

- Idle: Serial flash memory device not selected – no interaction with the serial flash memory device and the IOFx signals are driven.
- Instruction: Serial flash memory device selected – the instruction is sent to the serial device and all the IOFx signals are driven.
- Address: Serial flash memory address is sent to the device – all the IOFx signals are driven and this phase is not applicable for all SFM commands.
- Mode: Mode bytes are sent to the serial flash memory device – all the IOFx signals are driven and this phase is not applicable for all SFM commands.
- Dummy: Dummy clocks are provided to the serial flash memory device. See Figure 432 for the IOFx signals driven. The actual data lines required for the SFM command executed are not driven for data read commands.

NOTE

— This phase is not applicable for all the SFM commands.

— All read commands in Dual pad, Quad pad or Octal pad modes must use dummy phase before read phase. Note that this restriction is not applicable to Single-pad mode.

- Data: Serial flash memory data are sent to or received from the serial flash memory device. See the preceding figure for the IOFx signals driven. The actual data lines required for the SFM command executed are not driven for data read commands.

NOTE

This phase is not applicable for all the SFM commands.

The PCSFx and SCKFx signals are driven permanently throughout all the phases. In the individual flash memory mode, this applies to the selected flash memory device.

Access to a single, individual serial flash memory device

See Serial flash memory access schemes for details.

Read access to two serial flash memory devices attached to the QuadSPI module in parallel. See Serial flash memory access schemes for details.

74.4 Functional description

This section provides a functional description of the QuadSPI module.

74.4.1 Serial flash memory access schemes

In the individual flash memory mode, all supported commands are available.

74.4.2 Normal mode

This mode allows communication with an external serial flash memory device. Compared to the standard SPI protocol, this communication method uses up to four bidirectional data lines operating at high-data rates. The communication to the external serial flash memory device consists of an instruction code and optional address, mode, dummy, and data transfers. The flexible programmable core engine described below is immune to a wide variety of command or protocol differences in the serial flash memory devices provided by various flash memory vendors.

74.4.2.1 Programmable sequence engine

The core of the QuadSPI module is a programmable sequence engine that works on "instruction-operand" pairs. The core controller executes each programmed instruction sequentially. The complete list of instructions and the corresponding operands are provided in the following table.

Table 547. Instruction set

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
CMD	1d	N={0,1,2}d 0d - One pad 1d - Two pads 2d - Four pads	8-bit command value	Provide the serial flash memory with operand on the number of pads specified
ADDR	2d		Number of address bits to be sent (for example, 24d	Provide the serial flash memory with address cycles according to the operand on the number of pads specified

Table continues on the next page...

Table 547. Instruction set (continued)

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
			=> 24 address bits required)	The actual address to be provided is derived from the incoming address in case of AHB-initiated transactions and the value of SFAR in case of IPS-initiated transactions.
DUMMY	3d		Number of dummy clock cycles (should be <= 64 and > 2 cycles)	Provide the serial flash memory with dummy cycles according the operand The PAD information defines the number of pads in input mode. For example, one pad implies that pad 1 is not driven, rest all are driven.
MODE	4d		8-bit mode value	Provide the serial flash memory with 8-bit operand on the number of pads specified
MODE2	5d	N={0,1}d	2-bit mode value	Provide the serial flash memory with 2-bit operand on the number of pads[1] specified
MODE4	6d	N={0,1,2}d	4-bit mode value	Provide the serial flash memory with 4-bit operand on the number of pads[2] specified
READ	7d	N={0,1,2}d 0d - One pad 1d - Two pads 2d - Four pads	Read data size in bytes (for AHB transactions, your application should ensure that data size is a multiple of 8 bytes)	Read data from flash memory on the number of pads specified The data size can be overwritten by writing to the ADATSZ field of the BUFxCR registers for AHB-initiated transactions and to the IDATSZ field of the IP Configuration Register (IPCR) for IPS-initiated transactions.
WRITE	8		Write data size in bytes	Write data on the number of pads specified The data size can be overwritten by writing to the IDATSZ field of IP Configuration Register (IPCR).
JMP_ON_CS	9d	NA	Instruction number	Every time the CS is deasserted, jump to the instruction pointed to by the operand. This instruction allows the programmer to specify the behavior of the controller when a new read transaction is initiated following a CS deassertion.
STOP	0d	NA	NA	Stop execution; deassert CS

1. For a one-pad instruction, MODE2 takes two serial flash memory clock cycles on the flash memory interface.
2. For a one-pad instruction, MODE4 takes four serial flash memory clock cycles on the flash memory interface. For a four-pad instruction, MODE4 takes one serial flash memory clock cycle on the flash memory interface.

The programmable sequence engine allows you to configure the QuadSPI module according to the serial flash memory connected on board. This flexible structure is compatible with new command or protocol changes from different vendors.

74.4.2.2 Flexible read xAHB buffers

To reduce the latency of the reads for AHB masters, the data read from serial flash memory is buffered in flexible AHB buffers. There are four such flexible buffers. The size of each of these buffers is configurable with the minimum size being 0 bytes and maximum size being the size of the complete buffer instantiated (256 bytes). The size of buffer 0 ranges from 0 to BUF0IND. The size of buffer 1 ranges from BUF0IND to BUF1IND, buffer2 from BUF1IND to BUF2IND and, buffer 3 ranges from BUF2IND to the size of the complete buffer (256 bytes).

Each flexible AHB buffer is associated with the following:

- An AHB master: Optionally, buffer3 may be configured as an "all master" buffer by writing 1 to BUF3CR[ALLMST]. When buffer3 is configured in such a way, any access from a master not associated with any other buffer is routed to buffer3.
- A datasize field representing the amount of data to be fetched from the flash memory on every missed access.

The master port number of every incoming request is checked and the data is returned or fetched into the corresponding associated buffer. Every missed access to the buffer causes the controller to clear the buffer and fetch the BUFxCR[ADATSZ] amount of data from the serial flash memory. As such, you need not configure the buffer size to be greater than ADATSZ because the locations greater than ADATSZ are never used. For any AHB access, the sequence pointed to by BFGENCR[SEQID] is used for the initiated flash memory transaction. The data is returned to the master as soon as the requested amount is read from the serial flash memory. The controller; however, continues to prefetch the rest of the data in anticipation of a next consecutive request. See Figure 433 that shows flexible AHB buffers.

BFGENCR[SEQID] points to an index of the LUT. See LUT for details.

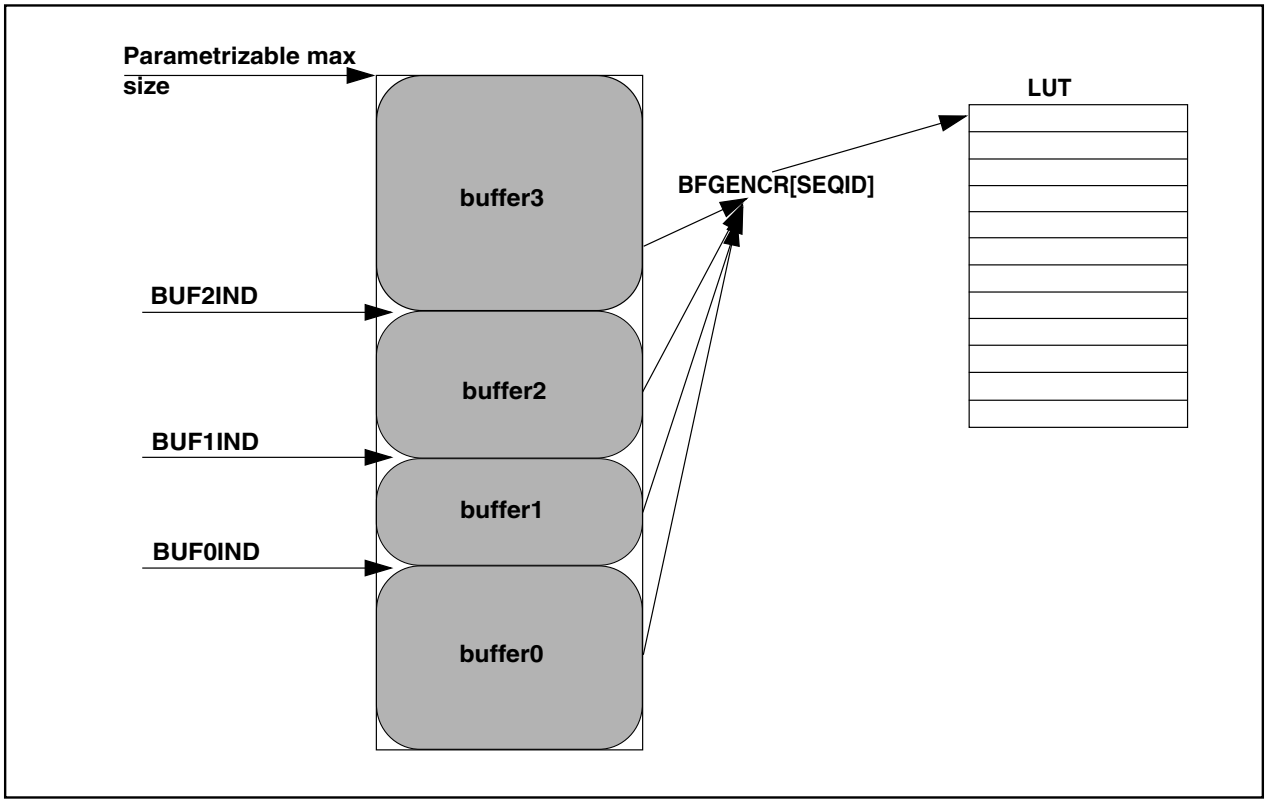


Figure 433. Flexible read AHB buffers

74.4.2.3 Abort mechanism during AHB read

Any ongoing read transaction is aborted if a request from the same master arrives for a location other than the location at which the transaction is going on. The abort can happen at any point of time.

74.4.2.4 HBURST support with AHB read

QuadSPI controller supports HBURST and HSIZE on the AHB read interface. HBURST indicates if the transfer forms part of a burst. Four, eight, and sixteen beat bursts are supported and the bursts might either be incrementing or wrapping. HSIZE indicates the size of the transfer, and supports 8-, 16-, 32-, and 64-bit data sizes. In case of WRAP accesses, QuadSPI generates aligned accesses to serial flash memory if there is no buffer hit for any incoming, non-sequential AHB read access. In case there is a buffer hit, the incoming address in the haddr line is latched as it is. If the total burst size is more than the data prefetch size, an error response is generated and the value of FR[AIBSEF] is configured as 1. The data prefetch size can be defined by BUFxCR[ADATSZ] or data size mentioned in the sequence pointed to by the SEQID field when ADATSZ is programmed as 0. In case of wrap burst, data prefetch size must be greater than or equal to the wrap burst size + 32 bytes. A few examples are shown in the figure below:

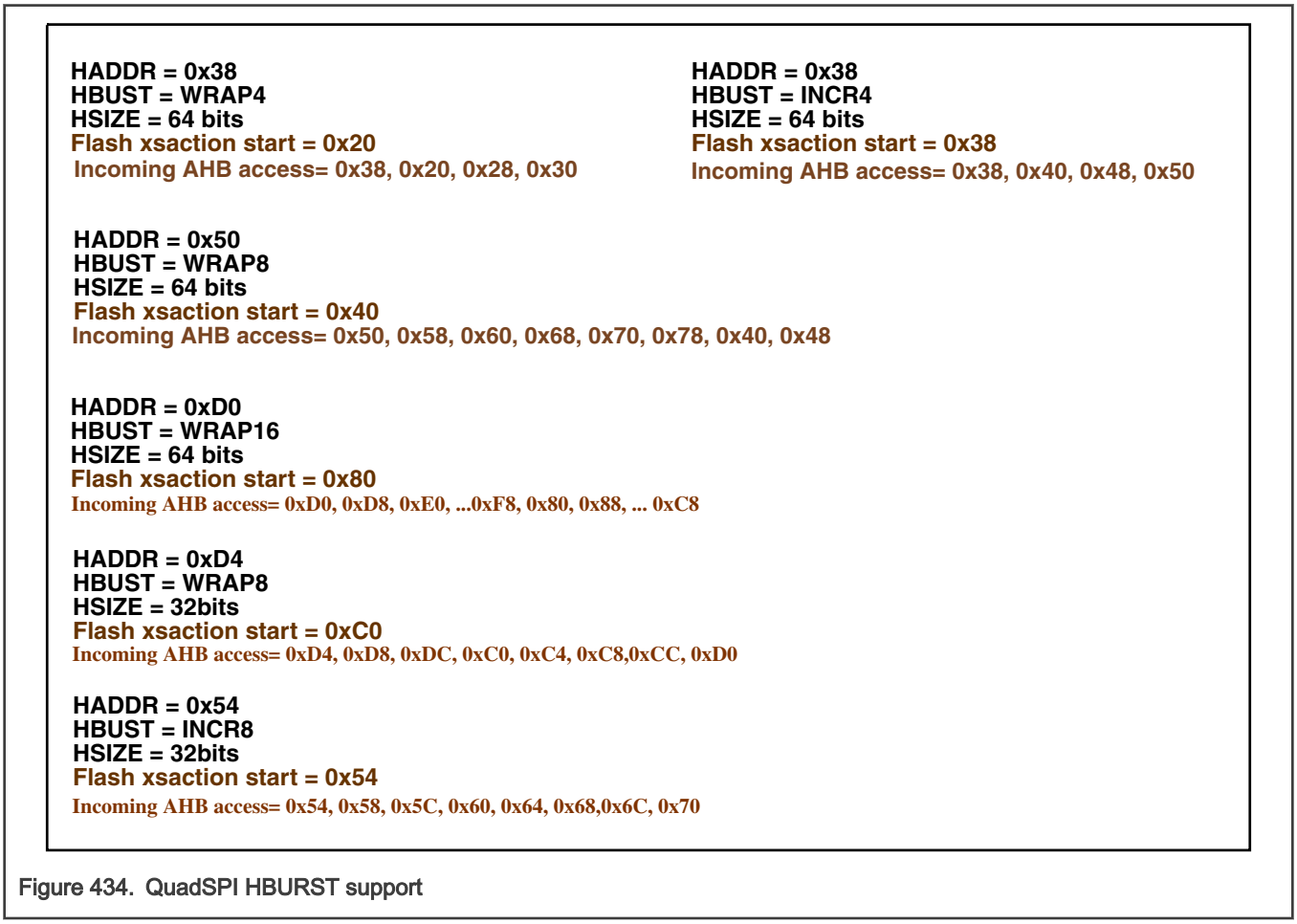


Figure 434. QuadSPI HBURST support

NOTE

The software must take care that the prefetch size should never be set less than the minimum data needed by any external interface to start processing.

NOTE

Whenever a core accesses QuadSPI memory with cache enabled, the prefetch size must be configured as equal or more than the cache line size; otherwise, FR[AIBSEF] error appears.

74.4.2.5 LUT

The LUT consists of a number of pre-programmed sequences. Each sequence is basically a sequence of instruction-operand pairs, which when executed sequentially, generate a valid serial flash memory transaction. Each sequence can have a maximum

of 10 instruction-operand pairs. The LUT can hold a maximum of sequences. The figure below shows the basic structure of the sequence in the LUT.

At reset, the index 0 of the LUT[0..4] is programmed with a basic read sequence as described in Reset sequence. After reset, the complete LUT may be reprogrammed according to the chip connected on board. To protect its contents, during a code runover, the LUT might be locked, after which a write to the LUT will not be successful until it has been unlocked again. The key for locking or unlocking the LUT is 5AF05AF0h, and the associated processes are as follows:

Locking the LUT

1. Write the key 5AF05AF0h into the LUT Key Register (LUTKEY).
2. Write 0b01 to the LUT Lock Configuration Register (LCKCR). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued). A successful write to this register locks the LUT.

Unlocking the LUT

1. Write the key 5AF05AF0h into the LUT Key Register (LUTKEY).
2. Write 0b10 to the LUT Lock Configuration Register (LCKCR). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued in between). A successful write to this register unlocks the LUT.

The lock status of the LUT can be read from the LCKCR[UNLOCK] and LCKCR[LOCK] fields.

Some example sequences are defined in Example sequences. The reset sequence at LUT index 0 is shown in the following table.

Table 548. Reset sequence

Instruction	Pad	Operand	Comment
CMD	0h	3h	Read data byte command on one pad
ADDR	0h	18h	24 address bits to be sent on one pad
READ	0h	8h	Read 64 bits
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

74.4.2.6 Issuing SFM commands

Each access to the external device follows this sequence:

1. You must pre-populate the LUT with the serial flash memory command sequences that are required for the flash memory device being used.
2. The module executes the instructions in this sequence one by one. The transaction starts and the module configues the value of SR[BUSY].
3. Communication with the external serial flash memory device starts and the transaction executes.
4. After the transaction is complete (all transmit and receive operations with the external serial flash memory device are complete), the module resets SR[BUSY]. In case of an IP command, FR[TFF] is asserted.

For details, see Flash memory programming and Flash memory read.

You can trigger the processing of SFM commands in the QuadSPI module in one of the following ways:

Using IP commands

For IP commands, the required components need to be written into the following registers and in this sequence:

1. Write the serial flash memory address to be used as provided in the Serial Flash Address Register (SFAR). For IP commands not related to specific addresses, the base address of the related flash memory needs to be programmed.

For example, for an instruction which does not require an address (that is, write enable instruction), the SFAR should be programmed with the base address of the memory the command is to be sent to.

2. Write the sequence ID and data size details in the IP Configuration Register (IPCR).
3. Note that writing a value to IPCR[SEQID] must be the last step of the sequence. It is possible to combine all the fields of the IPCR into one single write. See IP Configuration Register (IPCR) for details.

Using AHB commands

Any AHB memory-mapped access is routed to one of the buffers depending on the master port number of the request. If the access is a "miss," a new serial flash memory transaction is initiated. The transaction is based on the sequence pointed to by BFGENCR[SEQID] as described in Flexible read xAHB buffers.

An AHB access is considered memory mapped when the access is to the memory-mapped serial flash memories, as described in Memory Mapped Serial Flash Data - Individual Flash mode on Flash memory A.

74.4.2.7 Flash memory programming

In all cases, the memory sector to be written needs to be erased first. The programming sequence is then initiated in the following way:

1. Check that SR[BUSY] is de-asserted or the value of the BUSY field is 0, also, check that the TX buffer is empty. If you need to discard the data present in the TX buffer (SR[TXNE]) then the TX buffer must be cleared by writing 1 to MCR[CLR_TXF].
2. Program the address related to the command in SFAR.
3. Provide initial data for the program command into the circular buffer through the TBDR. At least one words of data must be written into the TX buffer up to a maximum of 32 entries.
4. Program the IPCR to trigger the command. IPCR[SEQID] should point to an index of the LUT that has the flash memory program sequence pre-programmed. You must write 1 to IPCR[IDATSZ] to denote the size of the write.
5. Repeat step 3, depending on the amount of data required, until all of the required data is written to the TBDR. SR[TXFULL] can be used to check if the buffer is ready to receive more data. At any time, TBSR[TRCTR] can be read to check how many words have been written into the TX buffer.

After writing to IPCR[SEQID] (see step 4), the module starts executing the programmed sequence. The software ensures that the correct sequence is programmed into the LUT in accordance with the flash memory connected to the module. The data is fetched from the TX buffer. It consists of 32 entries of 32-bit sizes and is organized as a circular FIFO, the read pointer for which is incremented after each fetch. When all the data is transmitted, the QuadSPI module returns from the busy state to the idle state. However, this is not true for the external device because the internal programming is still ongoing. You may monitor the relevant status information available from the serial flash memory device and ensure that the programming is done appropriately.

74.4.2.8 Flash memory read

Host access to the data stored in the external serial flash memory device is performed in two steps. First, the data must be read into the internal buffers and in the second step, these internal buffers can be read by the host.

Reading serial flash memory data into the QuadSPImodule internal buffers

A read access to the external serial flash memory device can be triggered in two different ways:

- IP command read: For reading flash memory data into the RX buffer, you must provide the correct sequence ID in IPCR[SEQID]. The sequence ID points to a sequence in the LUT. The software needs to ensure that a correct read sequence is programmed in the LUT in accordance with the serial flash memory device connected on board. You must program the SFAR , and IPCRs. All available read commands supported by the external serial flash memory are possible.

  Optionally, it is possible to clear the RX buffer pointer prior to triggering the IP command by writing a 1 to MCR[CLR_RXF].

  Using these inputs, the complete transaction is built when IPCR[SEQID] is written to. The transaction related to the read access starts and the requested number of bytes is fetched from the external serial flash memory device into the RX buffer. As the read access is triggered by an IP command, the value of both SR[IP_ACC] and SR[BUSY] is set to 1.. A count of the number of entries currently in the Rx buffer can be obtained from RBSR[RDBFL].

Communication with the external serial flash memory stops if the specified number of bytes are read (on successful completion of the transaction).

- AHB command read: For reading flash memory data into the AHB buffer, you must:
  - Set up a read access by a master to the address range in the system memory map, which the external serial flash memory devices are mapped to.
  -
  - Program the buffer registers corresponding to the AHB master initiating the request, and this depends on the configuration of RBCT[RXBRD].
  - Provide the correct sequence ID in the BFGENCR. The software ensures that a correct read sequence is programmed in the LUT in accordance with the serial flash memory device connected on board. Flash memory device selection and access mode are determined by the address accessed in the AHB address space associated with the QuadSPI module (see Memory-mapped serial flash memory data—individual flash memory mode on flash memory A

  On each AHB read access to the memory mapped area, the valid data in the AHB buffer is checked against the address requested in the actual read. When the AHB read request cannot be served from the content of the AHB buffer, the buffer is flushed and the controller executes the sequence pointed to by the sequence ID. The requested number of buffer entries defined in BUFxCR[ADATSZ] is then fetched from the external serial flash memory device into the internal AHB buffer. As the read access is triggered through the AHB bus, the value of SR[AHB_ACC] is set, driving SR[BUSY] in turn, until the transaction is complete. Communication with the external serial flash memory stops when the specified number of entries is filled.

Data transfer from the QuadSPI module internal buffers

The data read out from the external serial flash memory device by the QuadSPI module is stored in the internal buffers. The means of accessing the data from the buffer differs depending on which buffer the data is loaded to. See Block diagram for details on the two available buffers, the RX buffer and the AHB buffer, in this module. The buffer appears transparent to you and is non-memory mapped. See the "Flexible AHB Buffer" section for details.

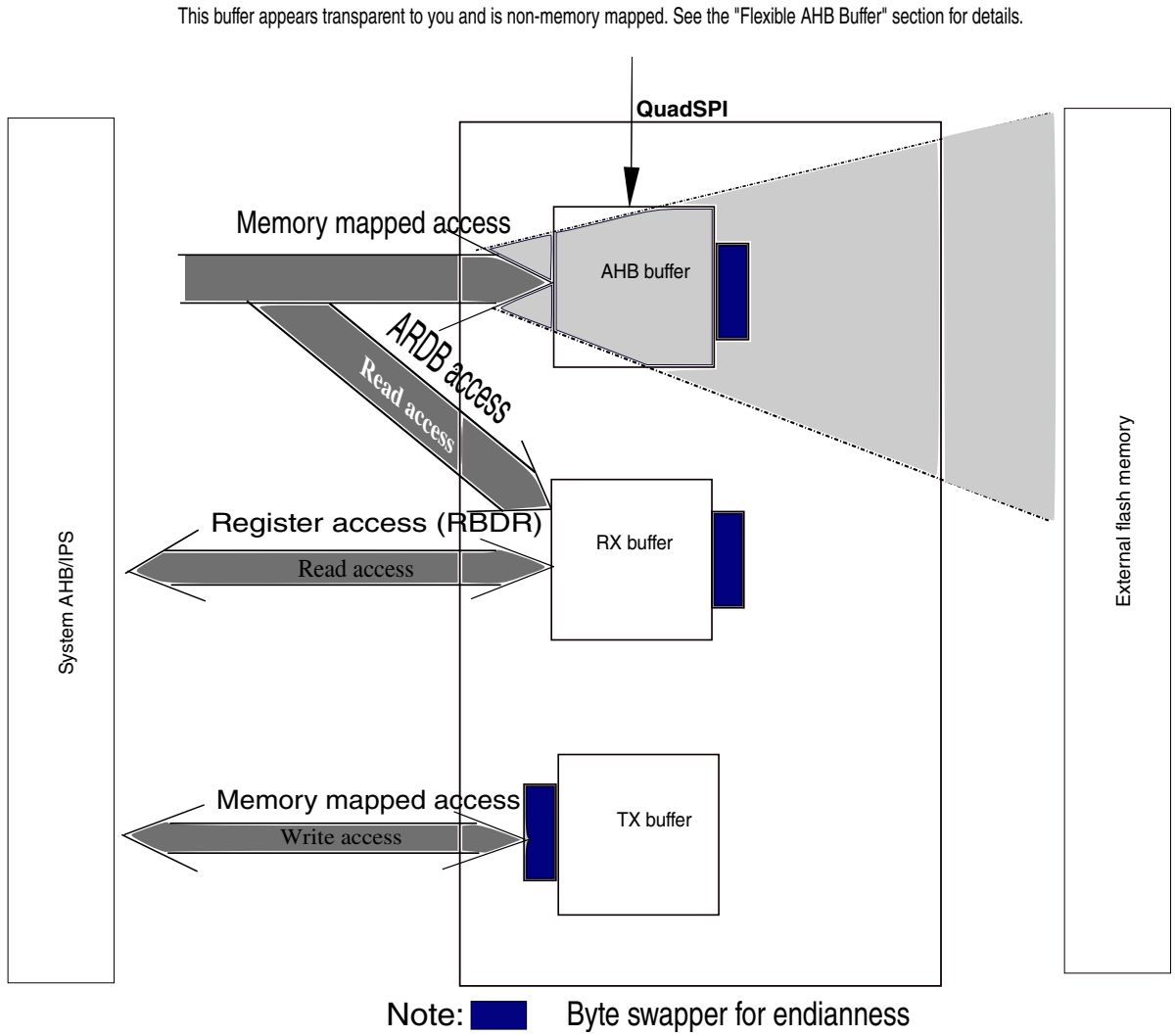


Figure 435. QuadSPI memory map

The RX buffer is implemented as FIFO of depth entries of 4 bytes. Its content is accessible in two different address areas, both referring to identical data and the same physical memory:

- In the IPS address space in the area associated with RX Buffer Data Register (RBDR0 - RBDR63).
- In the AHB address space in the area associated with AHB RX Data Buffer Register (ARDB0 - ARDB127).Two successive entries are accessed with one single 64-bit AHB read operation.

The RX buffer operation can be summarized as follows:

- RBCT[WMRK] determines at which fill level SR[RXWE] is asserted and how many entries are removed from the RX buffer on each buffer POP operation.
- SR[RXWE] indicates that the configured number of data entries is available in the RX buffer and RBSR[RDBFL] indicates how many valid entries are available in total.
- The first entry (RBDR0 or ARDB0) always corresponds to the first valid entry in the RX buffer.

For details, see RX Buffer Data Register (RBDR0 - RBDR63) and AHB RX Data Buffer Register (ARDB0 - ARDB127).

- Flag-based data read of the RX buffer is performed by polling SR[RXWE]. When it is asserted, the valid entries can be read either via the IPS address space (RBDRn) or the AHB address space (ARDBn). A buffer POP operation must be

triggered by the application by writing a 1 to FR[RBDF]. This automatically updates the FIFO to point to the next entry as defined by RBCT[WMRK]. For example, if WMRK is set to 3, then the buffer discards 16 bytes of data.

- DMA-controlled data read of the RX buffer is performed by using the DMA module. The application must ensure that the DMA controller of the related chip is programmed appropriately, as described in DMA usage.
- DMA-controlled read out is triggered fully automatically by the assertion of SR[RXWE]. The related buffer POP operation is also handled completely inside the QuadSPI module. As in the case explained here, accessing the RX buffer content either on RBDRn or ARDBn related addresses is equivalent.
- AHB buffer data read via memory-mapped access: This kind of access is performed by reading one of the addresses assigned to the external serial flash memory device(s) within the range specified in Table 570. If this is not the case, a memory-mapped AHB command read is triggered as described above. If the requested data is already available in the AHB buffer, it is provided directly to the host.

When an AHB access is made to the flash memory mapped address, the data is fetched and returned to the AHB interface. The AHB interface is stalled until the data is fetched. As soon as the data from the requested address is read by the QuadSPI module, the AHB read access is served. Therefore, it is possible to run sequential reads from the AHB buffer at arbitrary speed without the need to monitor any information about the availability of the data. Nevertheless, this access scheme stalls the AHB bus for the time required to read the data from the serial flash memory device. If you know that the access is sequential, a better way is to have a prefetch enabled by programming the value of BUFxCR[ADATSZ] so that the data is fetched into the buffer before the next sequential AHB access.

As long as the host restricts its accesses to the data present in the buffer and to the data currently fetched from the serial flash memory, it is possible to run the host read from the AHB buffer simultaneously with the serial flash memory read into the AHB buffer.

74.4.2.9 Byte ordering of serial flash memory read data

The basic scheme is that the first byte read out of the serial flash memory device, which is addressed by SFAR[SFADR], corresponds to RBDR0[31:24] for IP command read. In contrast to that for AHB command read, the bytes are always positioned according to the byte ordering of the AHB bus.

- Byte ordering in individual flash memory mode

  The following table provides the byte ordering scheme of how the byte oriented data space of the serial flash memory device is mapped into one single 32-bit entry of the RX buffer or the AHB buffer. The table is valid within the following context:

  — Flash memory A in individual flash memory mode

  — All AHB data read commands with 32-bit access size

Table 549. Byte ordering in individual flash memory mode

Serial flash memory byte numbering	3	2	1	0
Buffer entry bit position [31:0] (32-bit data width)	[31:24]	[23:16]	[15:8]	[7:0]

NOTE

For IP commands, the read size can be specified as number of bytes. If this number is not a multiple of four, then the last buffer entry is not completely filled with the missing higher numbered bytes at undefined values.

For AHB commands and reads, starting from an address not aligned to 32-bit boundaries, the requested bytes are given at the appropriate positions according to the AMBA AHB specification.

- Buffer entry ordering for 64-bit read access

  For read access via the AHB interface, a 64-bit access is possible. Each 64-bit access reads two 32-bit entries, simultaneously. The ordering of these 32-bit entries within the 64-bit word is provided in the following table.

Table 550. 64-bit read access buffer entry ordering

AHB read data bit position [63:0]	[63:32]	[31:0]
Buffer entry #	Odd (1, 3, 5, ...)	Even (0, 2, 4, ...)

74.4.2.10 Normal mode interrupt and DMA requests

The QuadSPI module has different flags that can only generate interrupt requests and one flag that can generate an interrupt as well as DMA requests. The following table lists the eight conditions. Note that the flags mentioned in the table are associated with the Flag Register (FR).

Table 551. Interrupt and DMA request conditions

Condition	Flag (FR)	DMA
TX buffer fill	TBFF	-
TX buffer underrun	TBUF	-
Illegal instruction error	ILLINE	-
RX buffer drain	RBDF	X
RX buffer overflow	RBOF	-
AHB buffer overflow	ABOF	-
AHB sequence error	ABSEF	-
AHB illegal transaction error	AITEF	-
AHB illegal burst size error	AIBSEF	-
IP command trigger during AHB access error	IPAEF	-
IP command trigger could not be executed error	IPIEF	-
IP command related transaction finished	TFF	-

Each condition has a corresponsing field in the Flag Register (FR) and a request enable field in the DMA Request Select and Enable Register (RSER). The FR[RBDF] has separate enable fields for generating IRQ and DMA requests. Note that not all the fields have an individual IRQ line. See the chip's Interrupt Vector table for details.

- Transmit buffer fill interrupt request

  This indicates that the TX buffer can accept new data. The buffer is asserted if FR[TBFF] is asserted and if the value of the corresponding enable field, RSER[TBFIE], is 1. See TX buffer Operation for details on the assertion of FR[TBFF].

  Apart from IRQ, it is possible to handle the TX buffer fill by using the DMA. If the value of RSER[TBFDE] is 1, a DMA request is triggered when the number of available space in the TX buffer is more than the TBCT[WMRK] valid entries and the value of SR[TXWA] is set. The application must configure the environment appropriately (for example, the DMA controller) for the DMA transfer.

- Receive buffer drain interrupt or DMA request

  This is derived from FR[RBDF], indicating that the RX buffer of the QuadSPI module has data available from the serial flash memory device to be read by the host. It remains set as long as RBSR[RXWE] is configured. Also, RSER[RBDIE] enables the related IRQ.

Apart from the IRQ, it is possible to handle the RX buffer drain by using the DMA. If the value of RSER[RBDDE] is 1, a DMA request is triggered when the RX buffer contains more than RBCT[WMRK] valid entries. The application must set the environment appropriately (for example, the DMA controller) for the DMA transfers.

- Buffer overflow/underrun interrupt request

  This is a combination of the following fields (all located in the Flag Register (FR) with the related enable bits in the DMA Request Select and Enable Register (RSER)):

  — TBUF - TX buffer underrun, enabled by TBUIE

  — RBOF - RX buffer overflow, enabled by RBOIE

  — ABOF - AHB buffer overflow, enabled by ABOIE

  — The transmit buffer underrun indicates that an underrun condition in the TX buffer has occurred. It is generated when a write instruction is triggered whilst the TX buffer is empty and the the value of RSER[TBUIE] is 1.

  — The receive buffer overflow indicates that an overflow condition in the RX buffer has occurred. It is generated when the RX buffer is full, an additional read transfer attempts to write into the RX buffer, and the value of RSER[RBOIE] is 1.

  — The AHB buffer overflow indicates that an overflow condition in the AHB buffer has occurred. It is generated when the AHB buffer is full, an additional read transfer attempts to write into the AHB buffer and the value of RSER[ABOIE] is 1.

  — The data from the transfers that generated the individual overflow conditions is ignored.

- Serial flash memory command error interrupt request

  If the IPAEF, IPIEF fields in the FR are set, and the related interrupt enable bits in the RSER are also set, then an interrupt is requested.

- Transaction finished interrupt request

  The IP command transaction finished IRQ indicates the completion of the current IP command. It is triggered by FR[TFF] and is masked by RSER[TFIE].

74.4.2.11 TX buffer operation

The TX buffer provides the data used for page programming. For proper operation, it is required to provide at least one entry in the TX buffer prior to starting the execution of the page programming command. The application must ensure that the required number of data bytes is written into the TX buffer fast enough as long as the command is executed without a TX buffer overflow or underrun.

The QuadSPI module sets the FR[TBFF] field as long as the TX buffer is not full and can accept more data. At the end of write through TX buffer, you must clear FR[TBFF] to avoid unnecessary last TX buffer fill interrupt. However, there would always be a pending request asserted from QuadSPI controller at the end of any DMA transfer. When external DMA finishes transfer iteration, this request from QuadSPI is kept asserted for the next iteration loop.

NOTE

Even if the generation of DMA requests for filling the TX buffer is disabled by using RSER[TBFDE], the TX buffer still accepts a DMA transfer because of the last asserted pending request.

Disabling of DMA transfer should be controlled by an external DMA master.

When the QuadSPI module tries to pull data out of an empty TX buffer, FR[TBUB] signals the TX buffer underrun. The TX buffer underrun flag is also asserted when the TX buffer contains less than 32-bits of data and the QuadSPI module tries to pull out data from it. The current IP command leading to the underrun condition is continued until the specified number of bytes is sent to the serial flash memory device. Also, the data that is transferred is in the Fs format. This means, after the underrun flag is set under this condition, it returns Fs until the required number of bytes are not sent. This has been done to ensure that the software does not erase the whole sector after underrun and just reprogramming from failure point serves the purpose. When this sequence command is complete, FR[TBFF] is asserted, indicating that the TX buffer is ready to be written again.

The TX buffer overflow is not signaled explicitly, but TBSR[TRBFL] can monitor the TX buffer fill level.

For more information, see TX Buffer Status Register (TBSR) and Flag Register (FR).

74.4.2.12 Address scheme

Earlier, serial flash memories supported only a 24-bit address space, restricting the maximum memory size of the serial flash memory to 16 MB. The new memory specification supports two types of 32-bit addressing mode in addition to the legacy 24-bit address mode.

Extended address mode

In this mode, the legacy 24-bit commands are converted to accept 32-bit address commands. The flash memory needs to be configured for the 32-bit address mode. Also, while programming the LUT sequence in QuadSPI for 32-bit mode, the ADDR commands should be programmed with 32d as the operand value,. By default, QuadSPI is in 24-bit legacy address mode. Each of the memory vendors have a different way of enabling this mode (see the memory specification from memory vendors). For example, the command B7h sent to the Macronix flash memory enables it for the 32-bit address mode.

Extended address register

In this mode, the upper 8 bits of the 32-bit register are provided by the Extended address register in the memory, which provides a specific register that is updated according to the address to be accessed. This effectively converts the legacy 24-bit address command into 32-bit address commands. The memories greater in size than 16 MB consist of banks of 16 MB each. The 8 bits occupied in the extended address register effectively enable a bank. For example, in Spansion memory, when the extended address register is updated with a value of 1h, with the help of the 17h command, it opens Bank1 of the memory. The consequent 24-bit address commands lead to Bank1. The extended address register needs to be updated with the respective value for access to other banks. This effectively converts the legacy 24-bit address command into 32-bit address commands.

This mode has been introduced for flash memories that need addresses segregated into rows and columns. The value in SFACR[CAS] defines the width of the column address required by a flash memory. The actual address to be provided is derived from the incoming address in case of AHB-initiated transactions and the value of SFAR in case of IPS-initiated transactions, if the value of SFACR[CAS] is 0. Otherwise, the actual address takes CAS into consideration. If the value of SFACR[CAS] is 3, then bits 26-3 of the programmed address are sent to the flash memory as its page address. This is in case the flash memory is operating in a 24-bit mode and bits 2-0 are sent as its column address. If a flash memory requirement for column address is less than the number of pads in which address has to be sent, then QuadSPI appends the remaining bits by 0. You must program the operand value in CADDR and CADDR_DDR command accordingly. It must be ensured that the total number of address bits requested by flash memory as its page and column address must not be more than 32 bits.

This mode has been introduced for flash memories, which have a word-addressable memory. This means, each address of the flash memory contains one word (two bytes) of data. The value of SFACR [WA] is configured to 1 to enter this mode. QuadSPI internally divides the incoming address in the AHB bus or the address in the SFAR to map it to a valid flash memory location. For example, if the incoming address is 2004h, the controller re-maps this address to access the flash memory location 1002h. If not in this mode, the incoming address 2004h is mapped to flash memory location 2004h.

74.4.3 Module Disable mode

Module Disable mode is a block-specific mode that the QuadSPI can enter to disable serial flash memory clock and AHB command. This mode can be entered by:

- The host software: by writing a '1' to MCR[MDIS]

Below are the condition that must be fulfilled to enter the Module Disable mode:

- SR[BUSY] = 0
- SR[AHBTRN] = 0
- RBSR[RDBFL] = 0
- SR[RXDMA] = 0
- SR[TXDMA] = 0
- None of the flags in FR are enabled as interrupts is set

The conditions mentioned above ensures the following:

- There is no SFM command currently being executed.
- All the data read into the RX buffer from the serial flash memory have been fetched by the application.
- There is no current AHB access.
- There is no active DMA request.
- There is no enabled interrupt that is pending.

Certain read or write operations have a different effect when the QuadSPI is in the Module Disable mode. In the Module Disable mode, not all of the status and flag bits of the QuadSPI module are updated, and writing to them has no effect. Interrupt and DMA request signals cannot be cleared while in the Module Disable mode.

NOTE

It is illegal to issue a new SFM command starting two clock cycles prior to raising the request of entering the Module Disable mode until the QuadSPI stays in this mode.

74.4.4 Leaving Module Disable mode

In the Module Disable mode, the serial flash memory clock and AHB command to the QuadSPI module are switched off.

After the QuadSPI has left this mode and has returned to Normal mode, the execution of the first SFM command is deferred until the clock to drive that part of the module related to the serial flash memory device is available. Depending upon the point in time when the first SFM command is triggered, the actual execution of the command starts with a delay, respective with the re-enabling of the flash memory clock signal.

74.5 Initialization/application information

This section provides the initialization and application information of the QuadSPI module.

74.5.1 Power up and reset

The serial flash memory devices connected to the QuadSPI module might require special voltage characteristics of their inputs during power up or reset. The application must ensure this.

74.5.2 Available status/flag information

This section provides an overview of the different flags and statuses available, and their interdependencies for different use cases. The SR and FR are the related registers.

74.5.2.1 IP commands

See IP Configuration Register (IPCR) for additional details not explicitly covered in this paragraph.

- IP commands—normal operation

  Writing to IPCR[SEQID] triggers the execution of a new IP command. Given that this is a legal command, SR[IPACC] and SR[BUSY] are asserted simultaneously, immediately after the execution starts.

  After the instruction on the serial flash memory device is complete, these field deassert and FR[TFF] is configured.

- IP commands—error situations

  See Overview_of_Error_Flags for details.

74.5.2.2 AHB commands

See the "Reading serial flash memory data into the QuadSPI module internal buffers" topic in theFlash Memory Read section for details.

- AHB commands—normal operation

Memory-mapped read access to a serial flash memory address not contained in the AHB buffer triggers the execution of an AHB command. Given that this is a legal command, SR[AHB_ACC] and SR[BUSY] are asserted simultaneously, immediately after the execution starts. After the instruction on the serial flash memory device is complete, these fields are deasserted.

- IP commands—error situations

  See Overview of FR error flags for details.

74.5.2.3 SFM commands

An SFM command consists of an instruction code and all other parameters (for example, size or mode bytes) needed for that specific instruction code. Triggering a command either initiates a transaction on the external serial flash memory or results in an error. See Table 552 for details on errors.

74.5.2.4 Overview of error flags

The following table provides an overview of the different error flags in the FR and additional error-related details.

Table 552. Overview of FR error flags

Error category	Error flag in FR	Command execution on serial flash memory device TFF behavior (in case of IP commands only)	Description
AHB error flag	ABOF	Flash memory transaction continues until it finishes	Set when the module tries to push data into the AHB buffer that exceeds the size of the AHB buffer. Occurs only because of the wrong programming of BUFxCR[ADATSZ].
AHB error flag	AIBSEF	Flash memory transaction is aborted	Total burst size of the AHB transaction is greater than prefetch data size.
AHB error flag	AITEF	Flash memory transaction is aborted	No response is generated from QuadSPI to AHB bus in case of illegal transaction. Also, the watchdog timer expires.
Miscellaneous error flag	ILLINE	Flash memory transaction aborted	Illegal instruction error flag - set when an illegal instruction is encountered by the controller in any of the sequences.
Command arbitration error	IPIEF	TFF not asserted in conjunction with that command	IP command error - caused when IP access is currently in progress (IP_ACC is set) and during: • Write attempt to IPCR register • Write attempt to SFAR register • Write attempt to RBCT register
Command arbitration error	IPAEF	TFF not asserted in conjunction with that command	• AHB command already running, another IP command could not be executed

Table continues on the next page...

Table 552. Overview of FR error flags (continued)

Error category	Error flag in FR	Command execution on serial flash memory device TFF behavior (in case of IP commands only)	Description
			• AHB command already running, write attempt to IPCR[SEQID]
Buffer-related error	RBOF	TFF is asserted on completion	• RX buffer overrun
Buffer-related error	TBUF	TFF is asserted on completion	• TX buffer underrun

Note that only the buffer-related errors are associated with a transaction on the external serial flash memory. All the other errors do not trigger an actual transaction.

74.5.2.5 IP bus and AHB access command collisions

There are two flags related to this topic, FR[IPAEF] and FR[IPIEF]. See the "Reading serial flash memory data into the QuadSPI module internal buffers" topic of the Flash Memory read section for a description of the flags.

74.5.3 Flash memory device selection

Regardless of the SFM command (IP or AHB), the access mode is selected by specifying the 32-bit address value for the following SFM command.

For IP commands, the access mode is selected with the address programmed into the SFAR register. See Serial Flash Address Register (SFAR) for details.

For AHB commands, the access mode is determined by the memory-mapped address. See AMBA Bus Register Memory Map for details.

74.5.4 DMA usage

For a complete description of the DMA module, see the related DMA Controller chapter. This section only provides QuadSPI-specific DMA usage details.

74.5.4.1 DMA usage in normal mode

74.5.4.1.1 Bandwidth considerations

Careful consideration of the throughput rate of the entire chain (serial flash memory -> AHB bus / IP bus -> DMA controller) involved in the read/write data process is essential for a proper operation. Such analysis must take into account not only the data rate provided by the serial flash memory but also the data rate of the AHB bus and the performance of the DMA controller in reading/writing data from/to the RX/TX buffer.

Two figures must match for a proper operation, which means that the data rate provided by the serial flash memory device must not exceed the average RX buffer readout data rate. Otherwise, the longer this state persists, it results into an RX buffer overflow. Similarly, the data consumed by the serial flash memory device must not exceed the average TX buffer fill rate. If this persists, it leads to an underrun.

AHB bus side (data read)

The total number of bus cycles for each DMA minor loop completion is added from the following components:

The following table provides certain examples for typical use cases:

Case 1: DMA needs to read 4 bytes from SRAM and provide to QuadSPI. It costs total four bus clock cycles. Then, DMA handshake adds additional six bus clock cycles, resulting in a total of [6 + 4 * (32/4) = 38] bus clock cycles.

Table 553. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 60 MHz bus clock frequency
3	16	6+(16/4)*4 = 22	~366ns
7	32	6+(32/4)*4 = 38	~633ns
11	48	6+(48/4)*4 = 54	~900ns
15	64	6+(64/4)*4 = 70	~1166ns

Case 2: DMA needs to read 32 bytes from SRAM and provide to QuadSPI. DMA handshake takes an additional six bus cycles, with 32 bytes DMA read from SRAM costs (8 + 3) core clock cycles. DMA writes 32 bytes to QuadSPI, takes 2 * (32/4) = 16 bus cycles with one additional CPU access to QuadSPI, costing two bus clock cycles. This results in a total 6 + (8+3)/2 + 2 * (32/4) +2 = 30 bus clock cycles.

Table 554. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop >	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 80 MHz bus clock frequency
3	16	6 + (4+3) /2 + (16/4)*2 + 2 = 20	~333ns
7	32	6 + (8+3) /2 + (32/4)*2 + 2 = 30	~500ns
11	48	6 + (4+3)/2*3 + (48/4)*2 + 2 = 44	~733ns
15	64	6 + (8+3) + (64/4)*2 + 2 = 51	~810ns

NOTE

This table figure represents an ideal scenario; actual performance depends on how the chip integrates DMA and QuadSPI modules.

Serial flash memory device side (data read)

The number of serial flash memory cycles can be determined in the following way:

- Number of serial flash memory clock cycles is required to read 4 bytes, corresponding to one RX buffer entry (setup of command and address not considered): , eight cycles for quad mode (SDR) instructions in individual flash memory mode, and so on.
- Overhead because of clock domain crossing: one cycle

The following table lists the number of clock cycles required to read the data from the serial flash memory corresponding to the different settings of RBCT[WMRK]:

Table 555. Access duration examples for serial flash memory side

RBCT[WMRK] setting	Num bytes per DMA loop [1]	Num SCKFx for 80 MHz SCKFx		Time duration of flash memory data readout for 80 MHz SCKFx (~12.5ns period)	
		IFM [2] quad	IFM quad DDR	IFM quad	IFM quad DDR
0	4	8	4	~100ns	~50ns
1	8	16	8	~200ns	~100ns
3	16	32	16	~400ns	~200ns
7	32	64	32	~800ns	~400ns

1. DMA loop refers to one minor loop completion that is equivalent to one major loop iteration.
2. Individual flash memory mode

NOTE

The table figure represents an ideal scenario; actual performance depends on how the chip integrates with DMA and QuadSPI modules.

A complementary example is when the watermark is set to be too high. In such a case, the time taken by the DMA to read out the RX buffer entries should be lesser than the time taken by the controller to push in the remaining entries in the buffer.

IPS bus side (data write)

The total number of bus cycles for each DMA minor loop completion are added from the following components:

- Overhead for each minor loop, given by DMA controller: assume 10 cycles
- Overhead because to clock domain crossing: assume two cycles
- Number of bus clock cycles required for 16 bytes (128-bit write size): assume four cycles (read/write sequence of DMA controller)

Note that the size of the minor loop is determined by the size of TBCT[WMRK]; therefore, the overhead specified above distributes among (TBCT[WMRK]+1) write accesses of 32-bit each.

The following table provides some examples for typical use cases:

Table 556. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop [1]	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 80 MHz bus clock frequency
3	16	12+4 = 16	~200ns
7	32	12+8 = 20	~250ns
11	48	12+12 = 24	~300ns
15	64	12+16 = 28	~350ns
19	80	12+20 = 32	~400ns

1. DMA loop refers to one minor loop completion that is equivalent to one.

NOTE

The table figure represents an ideal scenario; actual performance depends on how the chip integrates with DMA and QuadSPI modules.

Serial flash memory device side (data write)

The number of serial flash memory cycles can be determined in the following way:

- Number of serial flash memory clock cycles required to write 16 bytes, corresponding to four TX buffer entry (setup of command and address not considered): 32 cycles for quad SDR writes in individual flash memory mode.
- Overhead due to clock domain crossing: one cycle

The following table lists the number of clock cycles required to read the data from the serial flash memory corresponding to the different settings of TBCT[WMRK]:

Table 557. Access duration examples for serial flash memory side

TBCT[WMRK] setting	Num bytes per DMA loop [1]	Num SCKFx		Time duration for consuming data at flash memory interface 100 MHz SCKFx (10 ns period)[2]		Time for FIFO to get empty[3]	
		IFM [4] quad	[5] single IO SDR mode	IFM quad	PFM single IO SDR	IFM quad	PFM single IO SDR
3	16	32	64	320ns	640ns	2240ns	4480ns
7	32	64	128	640ns	1280ns	1920ns	3840ns
15	64	128	256	1280ns	2560ns	1280ns	2560ns
23	96	192	384	1920ns	3840ns	640ns	1280ns

1. DMA loop refers to one minor loop completion that is equivalent to one major loop iteration.
2. Not all flash memory devices support writes at 100 MHz. See the flash memory data sheet for the actual page program frequency supported.
3. The assumption for these timings is that the TX Fifo is full when the transaction is initiated
4. Individual flash memory mode
5. Parallel flash memory mode

NOTE

The tables mentioned above are only examples which must be correlated with the DMA in the system.

Considering the examples provided in the two tables above for TX FIFO, it is evident that depending on the relationship between the bus clock and serial flash memory clock frequencies, there are settings possible where the serial flash memory consumes data faster than the IPS bus can write data in TX buffer. In these cases, a TX buffer underrun situation occurs. To avoid TX buffer underrun, the data transaction size should be large enough.

74.5.5 Flash memory devices address mapping

QuadSPI is configured in Single mode for the supported flash memory port A

The sizes of the flash memory devices are mapped with the system memory space based on the configurations of the following registers:

- SFA1AD
- SFA2AD
- SFB1AD
- SFB2AD

The total memory region for the flash memory devices is mapped between QuadSPI_AMBA_BASE and TOP_ADDR_MEMB2 such that the corresponding CS is asserted based on SFA1AD, SFA2AD and SFB1AD register configurations.

74.5.5.1 Single mode

For single mode configuration, you must write the same value to SFB1AD and SFB2AD registers that you write to the SFA2AD register.

For dual-die flash memories, the values you write to SFB1AD and SFB2AD registers corresponds to the mapped top addresses of each die.

For single-die flash memories, you must write the same value to SFA2AD register that you write to the SFA1AD register.

Following is a programming example for single mode single-die flash memory:

- QuadSPI_AMBA_BASE - 1000_0000h
- SFA1AD[TPADA1] - 2000_0000h
- SFA2AD[TPADA2] - 2000_0000h
- SFB1AD[TPADB1] - 2000_0000h
- SFB2AD[TPADB2] - 2000_0000h

The following figure illustrates the memory mapping for single mode QuadSPI configuration.

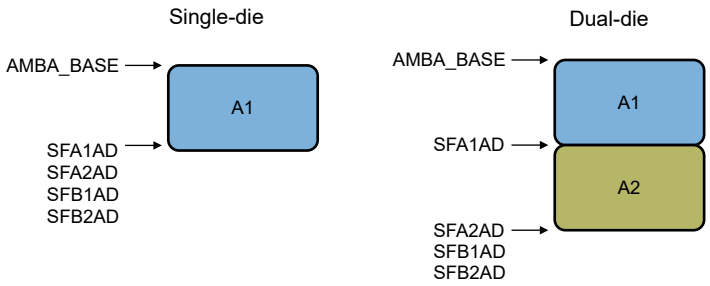


Figure 436. Memory map for Single mode

74.6 Byte ordering – endianness

The following topics show the byte ordering in 64-bit LE configuration for AHB buffer and 32-bit LE for TX/RX buffer.

74.6.1 Programming flash memory data

CPU writes instructions to the TBDR register, such as:

- Write TBDR: 4_03_02_01h
- Write TBDR: 8_07_06_05h

The following table shows the content against each TX buffer entry.

Table 558. Example of QuadSPI TX buffer

TX buffer entry	Content
0	4_03_02_01h
1	8_07_06_05h

Programming the TX buffer into the external serial flash memory device results in the following byte order to be sent to the serial flash memory:

- 01...02...03...04...05...06...07...08

74.6.2 Reading flash memory data into the RX buffer

Reading the content from the same address provides the following sequence of bytes, identical to the write case:

- 01...02...03...04...05...06...07...08

The following table shows the content against each TX buffer entry.

Table 559. Resulting RX buffer content

RX buffer entry	Content
0	4_03_02_01h
1	8_07_06_05h

74.6.2.1 Readout of the RX buffer through RBDRn

The RX buffer content appears at CPU read access through the peripheral bus interface in the following order:

- Read RBDR0: 4_03_02_01h
- Read RBDR1: 8_07_06_05h

74.6.2.2 Readout of the RX buffer through ARDBn

The RX buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- 32-bit access: Read ARDB0: 4_03_02_01h
- 32-bit access: Read ARDB1: 8_07_06_05h
- 64-bit access: Read ARDB0: 8_07_06_05_04_03_02_01h

74.6.3 Reading flash memory data into the AHB buffer

Reading the content from the same address as it was written to provides the following sequence of bytes, identical to the write case:

- 01...02...03...04...05...06...07...08

The following table shows the content against each TX buffer entry.

Table 560. Resulting AHB buffer content

AHB buffer entry	Content
0	8_07_06_05_04_03_02_01h

74.6.3.1 Readout of the AHB buffer through memory-mapped read

The AHB buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- 32-bit read access: 4_03_02_01h
- 32-bit read access: 8_07_06_05h
- 64-bit read access: 8_07_06_05_04_03_02_01h

74.7 Serial flash memory devices

Several different vendors make flash memory devices with a QuadSPI interface. At present, there is no set standard for the QuadSPI instruction set. The most common commands currently have the same instruction code for all vendors; however, some commands are unique to specific vendors. Some of the example sequences are provided in the following sections.

74.7.1 Example sequences

This section provides the example sequences of the QuadSPI module.

Table 561. Exit 4 x I/O read enhance performance mode (XIP) (Macronix) and read status

Instruction	Pad	Operand	Description
CMD	0h	EBh	4xIO read command
ADDR	2h	18h	24-bit address to be sent on four pads
MODE	2h	0h	2 mode cycles (exit XIP)
DUMMY	0h	4h	4 dummy cycles
READ	2h	8h	Read 64 bits
CMD	0h	5h	Read Status register
READ	0h	1h	Status register data
STOP	0h	0h	Stop, instruction over

74.7.1.1 Fast read sequence (Macronix/Numonyx/Spansion/Winbond)

The following table shows the fast read sequence for Macronix/Numonyx/Spansion/Winbond flash memories.

Table 562. Fast read sequence

Instruction	Pad	Operand	Description
CMD	0h	Bh	Fast read command = 0Bh
ADDR	0h	18h	24 address bits to be sent on one pad
DUMMY	0h	8h	Eight dummy cycles
READ	0h	4h	Read 32 bits on one pad
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

74.7.1.2 Fast read quad output (Winbond)

The following table shows the fast read quad output sequence for Winbond memories

Table 563. Fast read quad output sequence

Instruction	Pad	Operand	Description
CMD	0h	6Bh	Fast read quad output command = 6Bh
ADDR	0h	18h	24 address bits to be sent on one pad
DUMMY	0h	8h	Eight dummy cycles
READ	2h	4h	Read 32 bits on four pads
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

74.7.1.3 4 x I/O read enhance performance mode (XIP) (Macronix)

The following table shows the 4 x I/O read enhance performance mode for Macronix flash memories. The enhanced performance mode is also known as XIP mode.

Table 564. Fast read quad output sequence

Instruction	Pad	Operand	Description
CMD	0h	EBh	4xI/O read command = EBh
ADDR	2h	18h	24 address bits to be sent on four pads
MODE	2h	A5h	Two mode cycles
DUMMY	0h	4h	Four dummy cycles
READ	2h	4h	Read 32 bits on four pads
JMP_ON_CS	0h	1h	Jump to instruction 1 (ADDR)

When in XIP mode, the software must ensure that all the flash memories connected to the controller are in this mode. As a part of initializing the controller, all the flash memories might be enabled with XIP by carrying out dummy reads.

74.7.1.4 Dual command page program (Numonyx)

The following table shows the dual command page program sequence for Numonyx flash memories.

Table 565. Dual command page program sequence

Instruction	Pad	Operand	Description
CMD	1h	2h	Dual command page program = 02h on 2 pads
ADDR	1h	18h	24 address bits to be sent on two pads
WRITE	1h	20h	Write 32 bytes on two pads
STOP	0h	0h	Stop, instruction over

74.7.1.5 Sector erase (Macronix/Numonyx/Spansion)

The following table shows the Sector erase sequence for the Macronix/Numonyx/Spansion flash memories.

Table 566. Sector erase sequence

Instruction	Pad	Operand	Description
CMD	0h	20h	Sector erase command = 20h
ADDR	0h	18h	24 address bits to be sent on one pad
STOP	0h	0h	Stop, instruction over

74.7.1.6 Read status register (Macronix/Numonyx/Spansion/Winbond)

The following table shows the read status register sequence for Macronix/Numonyx/Spansion/Winbond flash memories.

Table 567. Read status register sequence

Instruction	Pad	Operand	Description
CMD	0h	0h5	Read status register command = 05h
READ	0h	0h1	Read status register data
STOP	0h	0h	Stop, instruction over

74.8 Sampling of serial flash memory input data

74.8.1 Basic description

QuadSPI is used to read data from the serial flash memory device. Depending on the actual implementation, there is a delay between the internal clocking in the QuadSPI module and the external serial flash memory device. See the following figure for an overview of this scheme.

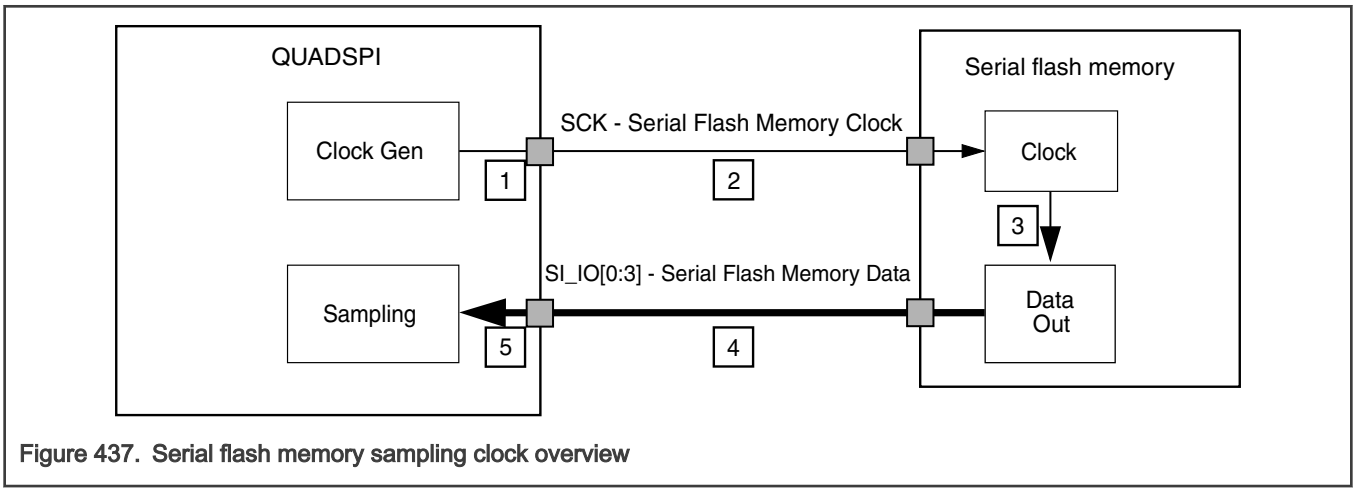


Figure 437. Serial flash memory sampling clock overview

The rising edge of the internal reference clock is taken as timing reference for the data output of the serial flash memory. After a time of $t_{Del,total}$ the data arrives at the internal sampling stage of the QuadSPI module. Considering the figure provided here, the following parts of the delay chain contribute to $t_{Del,total}$:

- Output delay of the serial flash memory clock output of the device containing the QuadSPI module
- Wire delay of application/PCB from the device containing the QuadSPI module to the external serial flash memory device
- Clock to data out delay of the external serial flash memory device, including input and output delays
- Wire delay of application/PCB from the external serial flash memory device to the device containing the QuadSPI module
- Device delay corresponding to the input data

NOTE

The $t_{Del,total}$ is specific to the characteristics of the actual implementation. Also, the serial flash memory device clock (SCK) is inverted with respect to the QuadSPI internal reference clock.

74.8.2 DQS sampling method

74.8.2.1 Basic description

In the DQS mode, the data strobe signal (DQS/RWDS) is used to sample the read data. Here, both DQS and the data sent by the flash memory move in the same direction; therefore, it is relatively easier to achieve at higher frequencies.

When using DQS for SDR reads, QuadSPI internally samples the incoming data on the rising edge of the strobe signal.

The next figure shows the sampling read data in the SDR mode using the DQS.

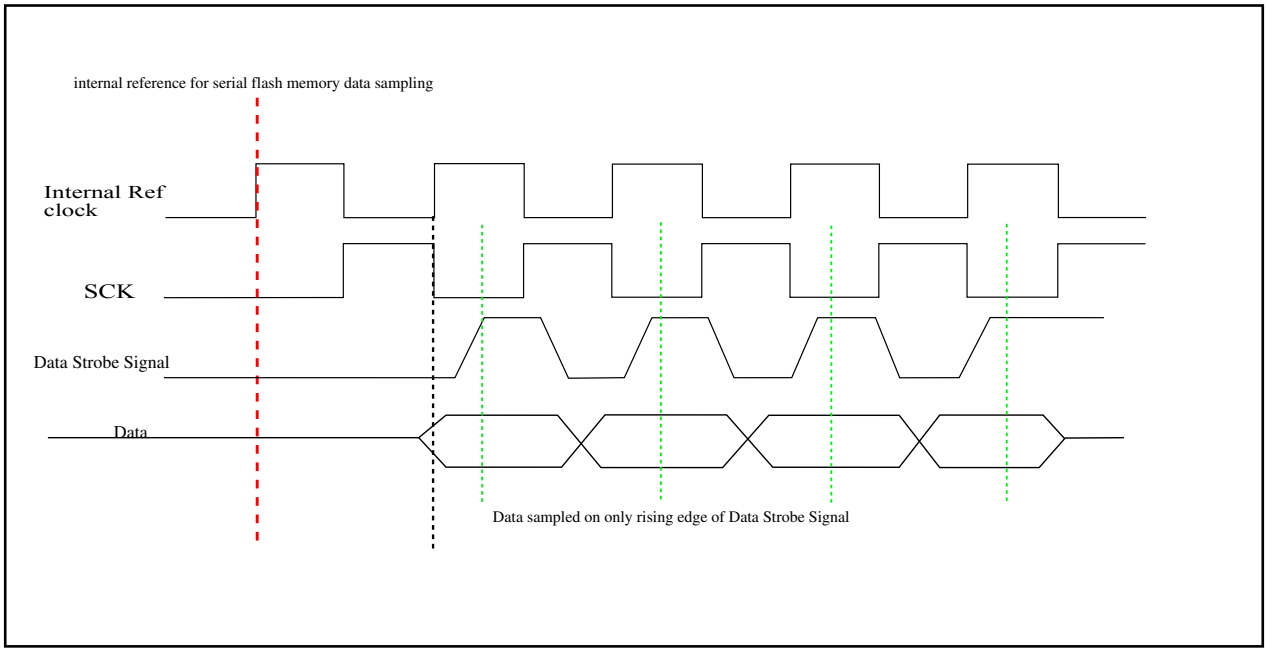


Figure 438. Data strobe functionality in SDR mode

74.8.2.2 Pad loopback

The internal clock is loop-backed from the dummy internal pad to compensate data pad delays. This can be enabled by configuring the value of MCR[DQS_FA_SEL] as "01" for flash memory A. This mode can be used with the following configuration:

- High/low frequency delay chain manual programming using DLLCRA[SLV_DLY_COARSE] or DLLCRB[SLV_DLY_COARSE]

NOTE

This mode may not be available on the chip. See the "Supported read modes" section in the chip-specific QuadSPI information for the read modes that this chip supports.

74.9 Delay chain usage

Slave delay chain programming sequence—

Following is the programming sequence for DLL bypass mode.

1. Program DLLCRA[SLV_EN]=1, DLLCRA[SLV_DLL_BYPASS]=1, and DLLCRA[SLAVE_AUTO_UPDT]=0.
2. Program the following fields to provide the desired DQS delay for sampling: DLLCRA[SLV_FINE_OFFSET], DLLCRA[SLV_DLY_COARSE], and DLLCR[FREQEN]. See the chip-specific QuadSPI information for the supported programming settings.
3. Program DLLCRA[SLV_UPD]=1 to load these values in the slave delay chain.
4. Check the slave delay chain update status by polling DLLSR[SLVA_LOCK]=1 and clear DLLCRA[SLV_UPD] after confirming the update state.

74.10 Memory map and register definition

This section provides the memory map and register definitions for the QuadSPI module.

74.10.1 Register write access

Following are the write access restriction terms that apply to all the registers:

- Register write access restriction

  For each register field, the write access conditions are specified in the detailed register description.

  The following table provides a description of the write access conditions. If, for a specific register bit or field, none of the given write access conditions is fulfilled, any write attempt to this register bit or field is ignored without any notification. The values of the bits or fields are not changed.

  The condition term [A or B] indicates that the register or field can be written to if at least one of the conditions is fulfilled.

Table 568. Register write access restrictions

Condition	Description
Anytime	No write access restriction
Disabled mode	Write access only if MCR[MDIS] = 1
Normal mode	Write access only if the module is in the normal mode

- Register write access requirements

  You can access all registers using 8-bit, 16-bit, and 32-bit wide operations. For some of the registers, at least a 16-bit or 32-bit wide write access is required to ensure correct operation. This write access requirement is stated in the detailed register description for each affected register.

74.10.2 QuadSPI register descriptions

This section provides the memory map and register definitions for the QuadSPI module.

Access to the following addresses does not result in a transfer error:

- 4h
- 50h
- 64h
- 104h
- 120h
- 138h
- 168h
- 188h
- 18Ch

74.10.2.1 QuadSPI memory map

QuadSPI base address: 404C_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration Register (MCR)	32	RW	000F_404Ch
8h	IP Configuration Register (IPCR)	32	RW	0000_0000h
Ch	Flash Memory Configuration Register (FLSHCR)	32	RW	0000_0303h
10h	Buffer 0 Configuration Register (BUF0CR)	32	RW	0000_0003h
14h	Buffer 1 Configuration Register (BUF1CR)	32	RW	0000_0002h
18h	Buffer 2 Configuration Register (BUF2CR)	32	RW	0000_0001h
1Ch	Buffer 3 Configuration Register (BUF3CR)	32	RW	8000_0000h
20h	Buffer Generic Configuration Register (BFGENCR)	32	RW	0000_0000h
24h	SOC Configuration Register (SOCCR)	32	RW	0000_0000h
30h	Buffer 0 Top Index Register (BUF0IND)	32	RW	0000_0000h
34h	Buffer 1 Top Index Register (BUF1IND)	32	RW	0000_0000h
38h	Buffer 2 Top Index Register (BUF2IND)	32	RW	0000_0000h
60h	DLL Flash Memory A Configuration Register (DLLCRA)	32	RW	0120_0000h
100h	Serial Flash Memory Address Register (SFAR)	32	RW	0000_0000h
108h	Sampling Register (SMPR)	32	RW	FF00_0000h
10Ch	RX Buffer Status Register (RBSR)	32	R	0000_0000h
110h	RX Buffer Control Register (RBCT)	32	RW	0000_0000h
134h	Data Learning Status Flash Memory A Register (DLSR_FA)	32	R	0000_0000h
150h	TX Buffer Status Register (TBSR)	32	R	0000_0000h
154h	TX Buffer Data Register (TBDR)	32	RW	0000_0000h
158h	TX Buffer Control Register (TBCT)	32	RW	0000_0000h
15Ch	Status Register (SR)	32	R	0200_3800h
160h	Flag Register (FR)	32	RW	0800_0000h
164h	Interrupt and DMA Request Select and Enable Register (RSER)	32	RW	0000_0000h
16Ch	Sequence Pointer Clear Register (SPTRCLR)	32	RW	0000_0000h
180h	Serial Flash Memory A1 Top Address Register (SFA1AD)	32	RW	7000_0000h
184h	Serial Flash Memory A2 Top Address Register (SFA2AD)	32	RW	7000_0000h
188h	Serial Flash Memory B1 Top Address Register (SFB1AD)	32	RW	7000_0000h
18Ch	Serial Flash Memory B2 Top Address Register (SFB2AD)	32	RW	7000_0000h
200h - 2FCh	RX Buffer Data Register (RBDR0 - RBDR63)	32	R	0000_0000h
300h	LUT Key Register (LUTKEY)	32	RW	5AF0_5AF0h
304h	LUT Lock Configuration Register (LCKCR)	32	RW	0000_0002h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
310h	LUT Register (LUT0)	32	RW	0818_0403h
314h	LUT Register (LUT1)	32	RW	2400_1C08h
318h - 35Ch	LUT Register (LUT2 - LUT19)	32	RW	0000_0000h

74.10.2.2 Module Configuration Register (MCR)

Offset

Register	Offset
MCR	0h

Function

This register holds configuration data associated with the QuadSPI operation.

Special write-access is permitted in different modes:

- DQS_FA_SEL: Disabled mode
- All other fields: Anytime

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved		Reserved	Reserved	DQS_FA_SEL		Reserved	0		0	Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	MDIS	Reserved	Reserved	0	0	0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved		SWRS THD	SWRS TSD
W					CLR_TXF	CLR_RXF										
Reset	0	1	0	0	0	0	0	0	0	1	0	0	1	1	0	0

Fields

Field	Function
31 —	Reserved
30	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
29-28 —	Reserved
27 —	Reserved
26 —	Reserved
25-24 DQS_FA_SEL	DQS clock for sampling read data at flash memory A Selects DQS clock for sampling read data at flash memory A QuadSPI port 00b - Reserved 01b - Pad loopback 10b - Reserved 11b - Reserved
23 —	Reserved
22 —	Reserved
21-20 —	Reserved
19-16 —	Reserved This field is reserved and should always be set to 0xF.
15 —	Reserved
14 MDIS	Module disable Allows the clock to the non-memory mapped logic in the QuadSPI to be stopped. 0b - Enable QuadSPI clocks 1b - Allow external logic to disable QuadSPI clocks
13 —	Reserved
12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11 CLR_TXF	Clear TX FIFO/buffer This is a self-clearing field that invalidates the TX buffer content. 0b - No action 1b - Read and write pointers of the TX buffer are reset to 0 and TBSR[TRCTR] is reset to 0.
10 CLR_RXF	Clear RX FIFO This is a self-clearing field that invalidates the RX buffer content. 0b - No action 1b - Read and write pointers of the RX buffer are reset to 0 and RBSR[RDBFL] is reset to 0.
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3-2 —	Reserved
1 SWRSTHD	Software reset for AHB domain 0b - No action 1b - AHB domain flops are reset. This field does not reset configuration registers. It is advisable to reset both the serial flash memory domain and AHB domain at the same time. Resetting only one domain might lead to side effects.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The software resets need the clock to be running to propagate to the design. The value of MCR[MDIS] should be 0 when the software reset bits are asserted. Also, before they can be deasserted again (by setting MCR[SWRSTHD] to 0), it is recommended to set the value of MCR[MDIS] to 1. After the software resets have been deasserted, the normal operation can be started by setting MCR[MDIS] to 0. **NOTE** Software must wait for at least three system cycles and three flash cycles after changing the value of this field.
0 SWRSTSD	Software reset for serial flash memory domain 0b - No action 1b - Serial flash memory domain flops are reset. This field does not reset configuration registers. It is advisable to reset both the serial flash memory domain and AHB domain at the same time. Resetting only one domain might lead to side effects. **NOTE** The software resets need the clock to be running to propagate to the design. The value of MCR[MDIS] should therefore be 0 when the software reset bits are asserted. Also, before they can be deasserted again (by specifying 0 as the value for MCR[SWRSTSD]), it is recommended to specify 1 as the value for MCR[MDIS]. After the software resets are deasserted, the normal operation can be started by specifying 0 as the value for MCR[MDIS]. **NOTE** Software must wait for at least three system cycles and three flash cycles after changing the value of this field.

74.10.2.3 IP Configuration Register (IPCR)

Offset

Register	Offset
IPCR	8h

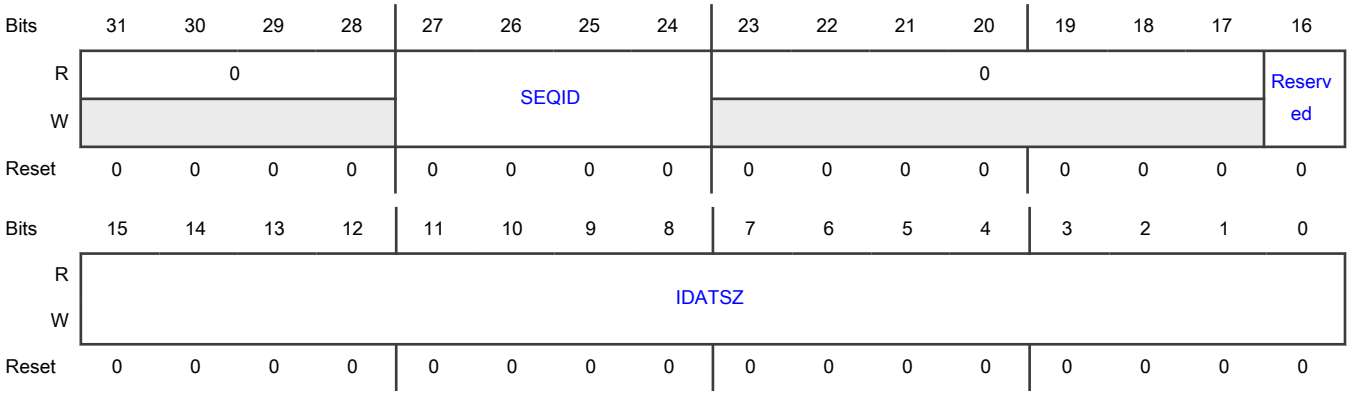
Function

This register provides all the configuration required for an IP-initiated command, which can be triggered by writing in the SEQID field of this register. If the SEQID field is written successfully, a new command to the external serial flash memory is initiated per the sequence pointed to by this field. See Normal mode for details on command triggering and command execution.

Special write-access is permitted if:

- SR[IP_ACC]=0

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 SEQID	Points to a sequence in the LUT This field defines bits [6:2] of the LUT index. The bits [1:0] are always assumed to be 0. See LUT for details. A write to this field triggers a transaction on the serial flash memory interface.
23-17 —	Reserved
16 —	Reserved
15-0 IDATSZ	IP data transfer size This field defines the data transfer size, in bytes, of the IP command.

74.10.2.4 Flash Memory Configuration Register (FLSHCR)

Offset

Register	Offset
FLSHCR	Ch

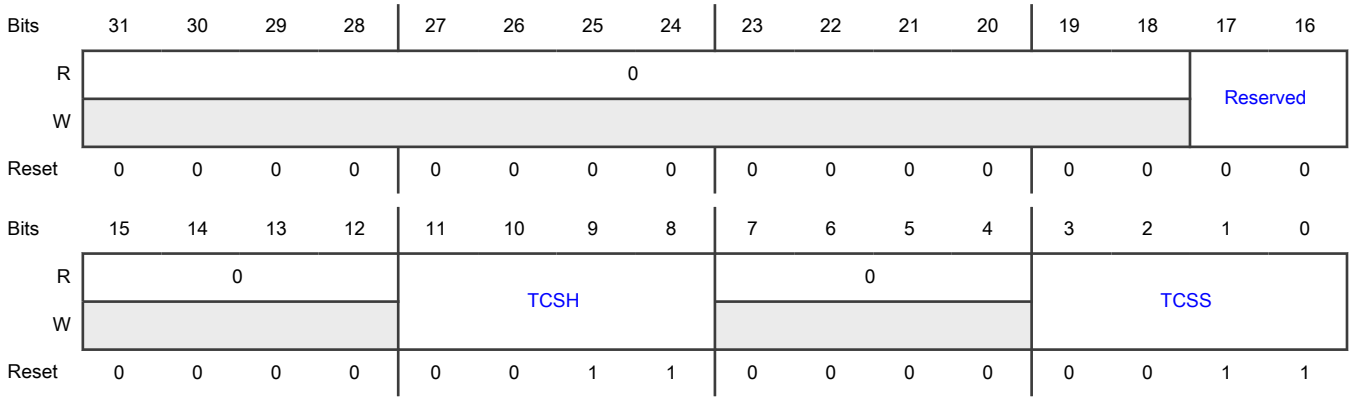
Function

This register contains the timings that are specific to the flash memory device. The QuadSPI controller must meet these timings for the device to function correctly.

Special write-access is permitted if:

- SR[AHB_ACC] = 0
- SR[IP_ACC] = 0

Diagram



Fields

Field	Function
31-18 —	Reserved
17-16 —	Reserved
15-12 —	Reserved
11-8 TCSH	Serial flash memory CS hold time This hold time is in terms of serial flash memory clock cycles, and it must be greater than or equal to five flash memory clock cycles.
7-4 —	Reserved
3-0 TCSS	Serial flash memory CS setup time This setup time is in terms of serial flash memory clock cycles, and it must be greater than or equal to two flash memory clock cycles.

74.10.2.5 Buffer 0 Configuration Register (BUF0CR)

Offset

Register	Offset
BUF0CR	10h

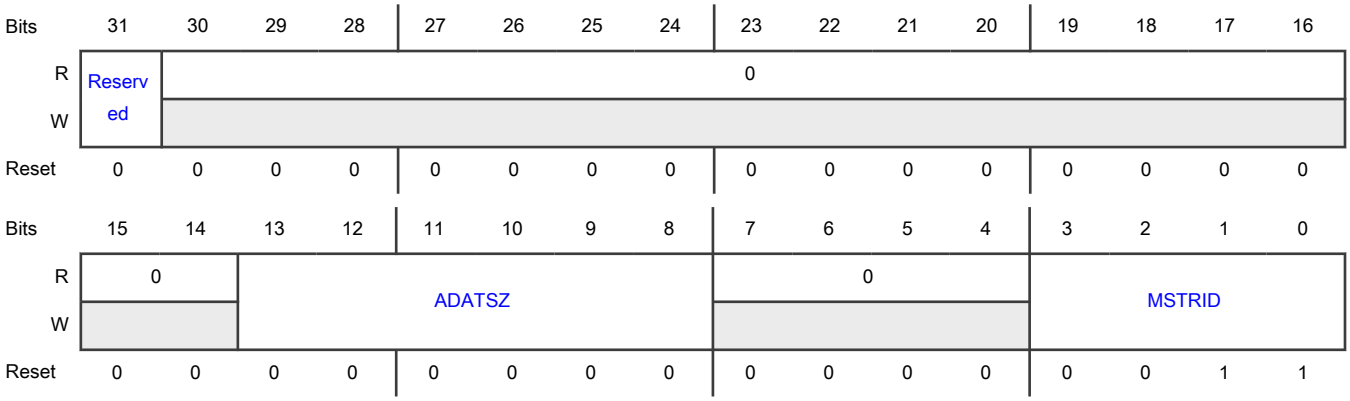
Function

This register provides the configuration for any read access routed to buffer0, which happens when the master port number of the incoming AHB request matches BUF0CR[MSTRID]. Any buffer "miss" leads to a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31 —	Reserved
30-14 —	Reserved
13-8 ADATSZ	AHB data transfer size Defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. For example, a value of 0x2 sets transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. The software should ensure that this transfer size is not greater than the size of the buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 0 Any AHB read access with this master port number is routed to this buffer. You must ensure that the master IDs associated with all buffers are different.

74.10.2.6 Buffer 1 Configuration Register (BUF1CR)

Offset

Register	Offset
BUF1CR	14h

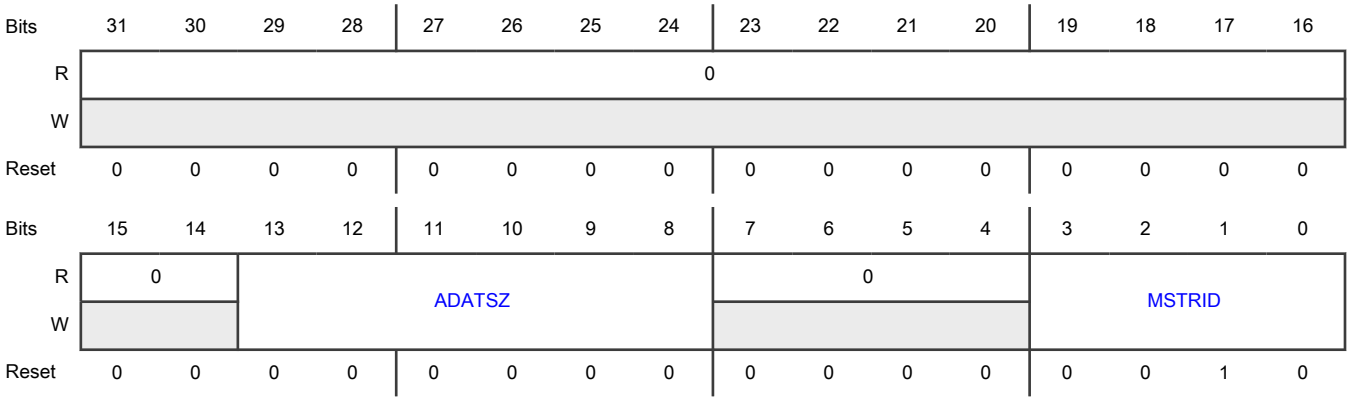
Function

This register provides the configuration for any access routed to buffer 1, which happens when the master port number of the incoming AHB request matches the MSTRID field of this register. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-14 —	Reserved
13-8 ADATSZ	AHB data transfer size This field defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. For example, a value of 0x2 sets the transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any AHB read access with this master port number is routed to this buffer. You must ensure that the master IDs associated with all buffers are different.

74.10.2.7 Buffer 2 Configuration Register (BUF2CR)

Offset

Register	Offset
BUF2CR	18h

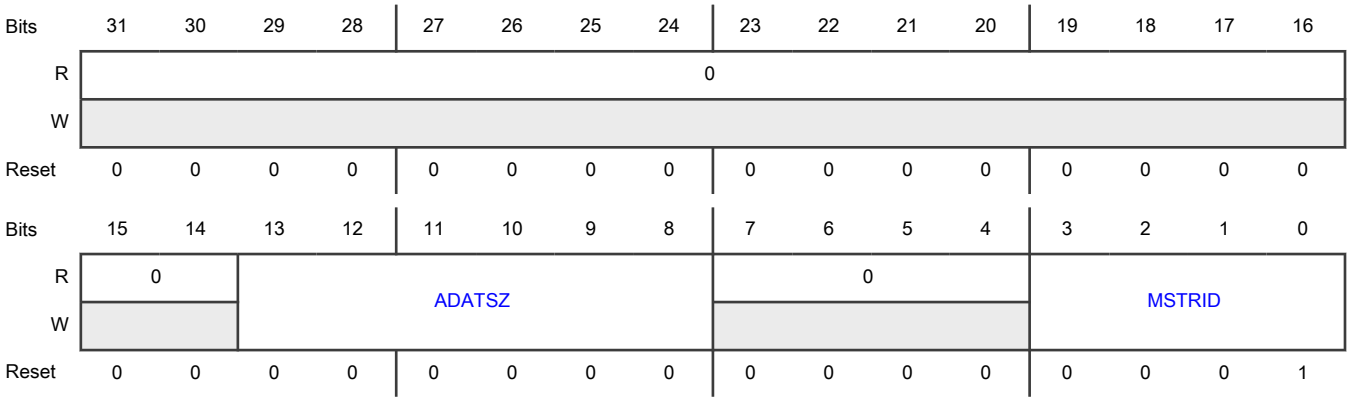
Function

This register provides the configuration for any access routed to buffer 2, which happens when the master port number of the incoming AHB request matches the MSTRID field of this register. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-14 —	Reserved
13-8 ADATSZ	AHB data transfer size This field defines the read data transfer size in 8 bytes of an AHB triggered read access to the serial flash memory. For example, a value of 0x2 sets transfer size to 16 bytes. When ADATSZ = 0, the data size

Table continues on the next page...

Table continued from the previous page...

Field	Function
	mentioned in the sequence pointed to by the SEQID field overrides this value. The software should ensure that this transfer size is not greater than the size of this buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID The ID of the AHB master associated with BUFFER2. Any AHB read access with this master port number is routed to this buffer. It must be ensured that the master IDs associated with all buffers are different.

74.10.2.8 Buffer 3 Configuration Register (BUF3CR)

Offset

Register	Offset
BUF3CR	1Ch

Function

This register provides the configuration for any access to buffer 3.

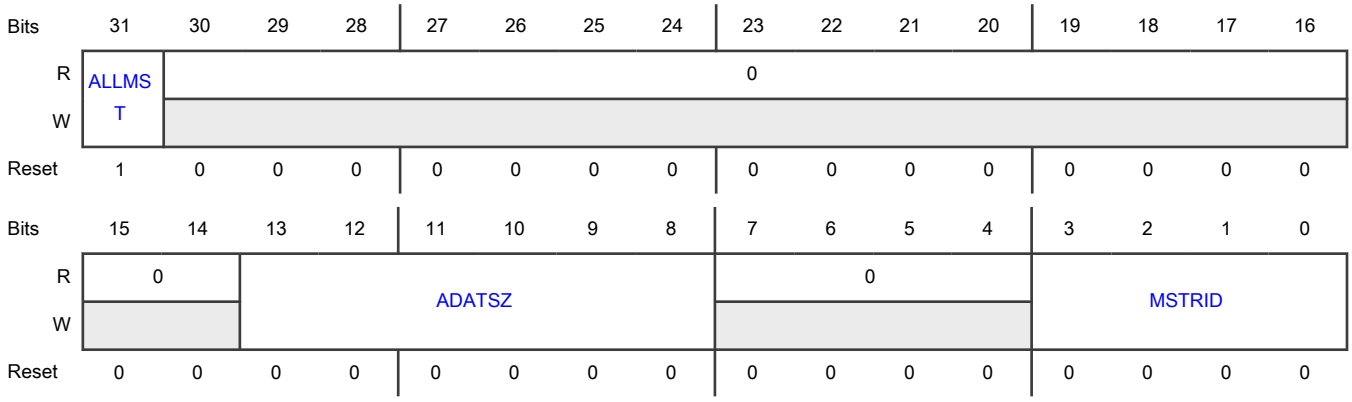
An access is routed to buffer 3 when the master port number of the incoming AHB request matches the MSTRID field of BUF3CR. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

In case the value of the ALLMST field is not 1, any such transaction (where master port number does not match any of the MSTRID fields) is returned with an ERROR response.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31 ALLMST	All master enable When set, buffer3 acts as an all-master buffer. Any AHB access with a master port number not matching with the master ID of buffer0, buffer1, or buffer2 is routed to buffer3. When set, the MSTRID field of this register is ignored.
30-14 —	Reserved
13-8 ADATSZ	AHB data transfer size Defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. The software should ensure that this transfer size is not greater than the size of this buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 3. Any AHB read access with this master port number is routed to this buffer. You must ensure that the master IDs associated with all buffers are different.

74.10.2.9 Buffer Generic Configuration Register (BFGENCR)

Offset

Register	Offset
BFGENCR	20h

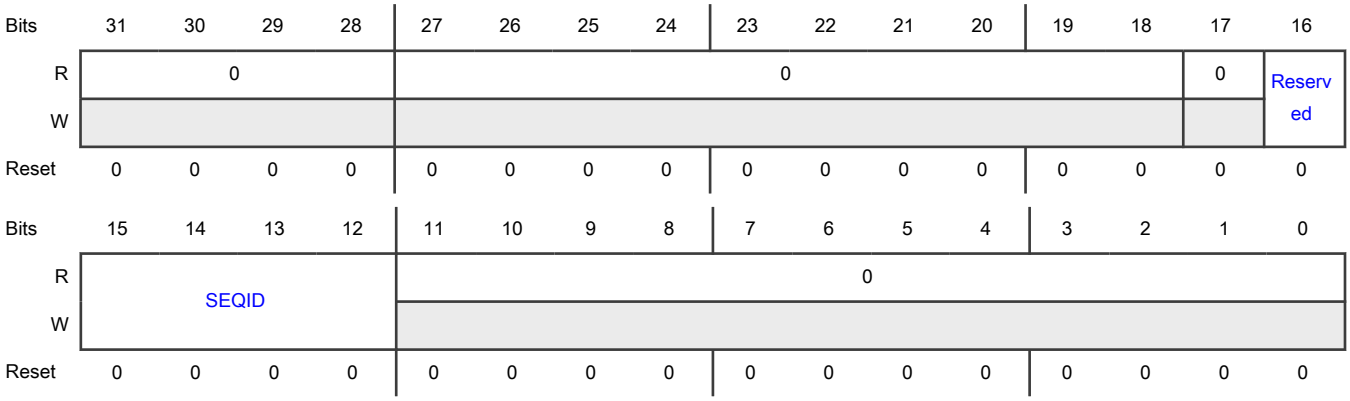
Function

This register provides generic configuration to any of the buffer accesses. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by the SEQID field.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-28 —	Reserved
27-18 —	Reserved
17 —	Reserved
16 —	Reserved
15-12 SEQID	Points to a sequence in the LUT This field defines the bits [6:2] of the LUT index. The bits [1:0] are always assumed to be 0. See LUT. **NOTE** If the sequence pointer differs in the new and the previous sequences, you should reset it. See sequence pointer clear register for more information.
11-0 —	Reserved

74.10.2.10 SOC Configuration Register (SOCCR)

Offset

Register	Offset
SOCCR	24h

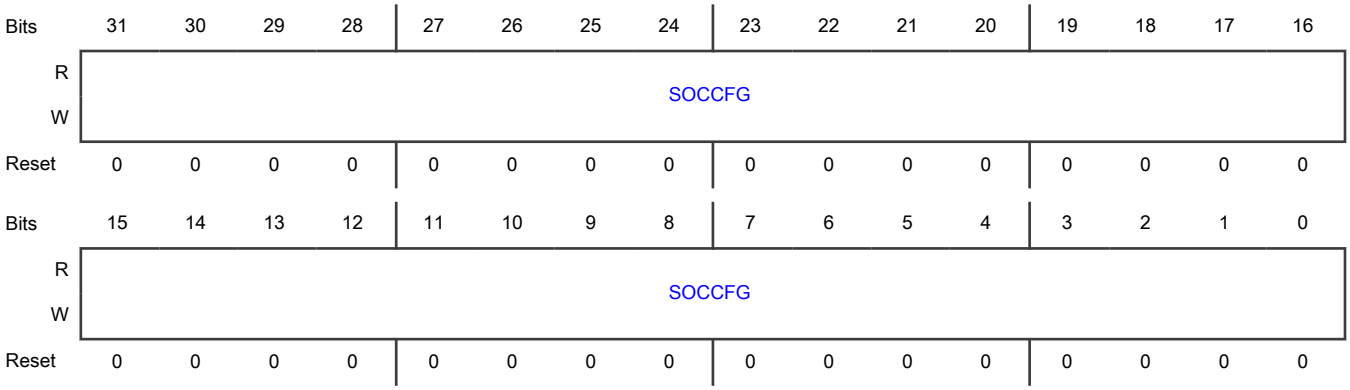
Function

This register is programmed at the chip level for QuadSPI delay chain configuration. For details, see chip-specific QuadSPI information.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-0 SOCCFG	SOC configuration This field configuration is specific to chip. For details, see chip-specific QuadSPI information.

74.10.2.11 Buffer 0 Top Index Register (BUF0IND)

Offset

Register	Offset
BUF0IND	30h

Function

This register specifies the top index for buffer 0, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The register value should be set to the desired number of bytes. For example, setting BUF0IND[31:3] to 0 gives 0 bytes, setting the value to 1 gives 8 bytes, and so on.

The size of buffer 0 is the difference between BUF0IND and 0.

The software must ensure that the value of TPINDX0 is not greater than the size of buffer 0.

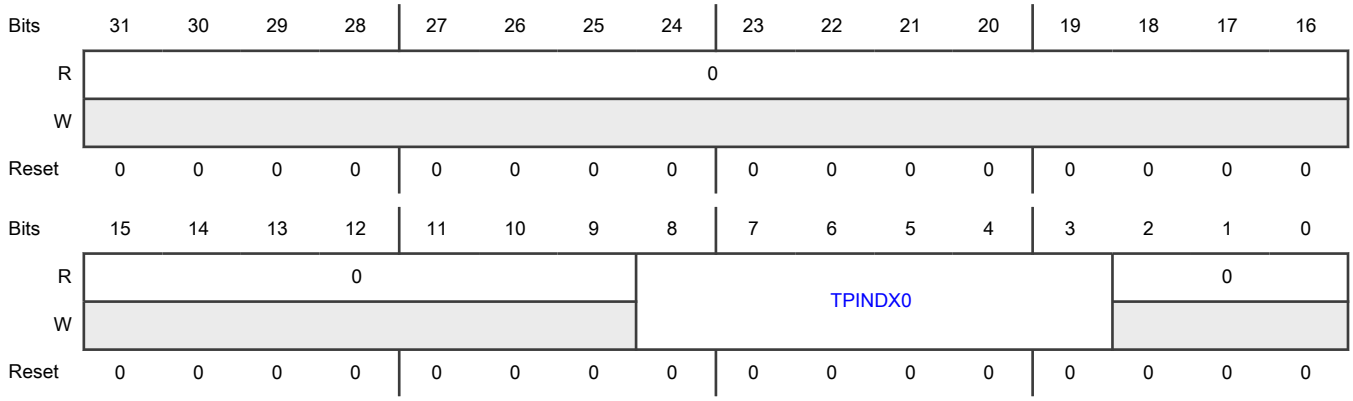
NOTE

The hardware does not provide any protection against illegal programming.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-9 —	Reserved
8-3 TPINDX0	Top index of buffer 0
2-0 —	Reserved

74.10.2.12 Buffer 1 Top Index Register (BUF1IND)

Offset

Register	Offset
BUF1IND	34h

Function

This register specifies the top index of buffer 1, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The size of buffer 1 is the difference between BUF1IND and BUF0IND. The register value should be entered in bytes. For example, if BUF0IND = 0x100, then setting BUF1IND = 0x130 sets the size of buffer 1 to 0x30 bytes.

The software must ensure that the value of TPINDX1 is not greater than the size of buffer 1.

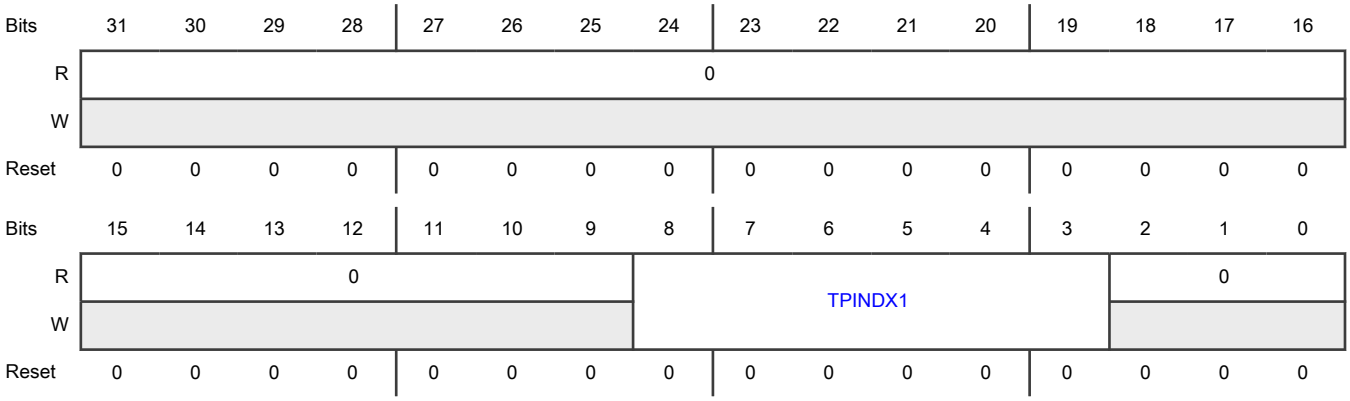
NOTE

The hardware does not provide any protection against illegal programming.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-9 —	Reserved
8-3 TPINDX1	Top index of buffer 1
2-0 —	Reserved

74.10.2.13 Buffer 2 Top Index Register (BUF2IND)

Offset

Register	Offset
BUF2IND	38h

Function

This register specifies the top index of buffer 2, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The size of buffer 2 is the difference between BUF2IND and BUF1IND. The register value should be entered in bytes. For example, if BUF1IND = 0x130 then setting BUF2IND = 0x180 sets the size of buffer 2 to 0x50 bytes.

The software must ensure that the value of TPINDX2 is not greater than the size of buffer 2.

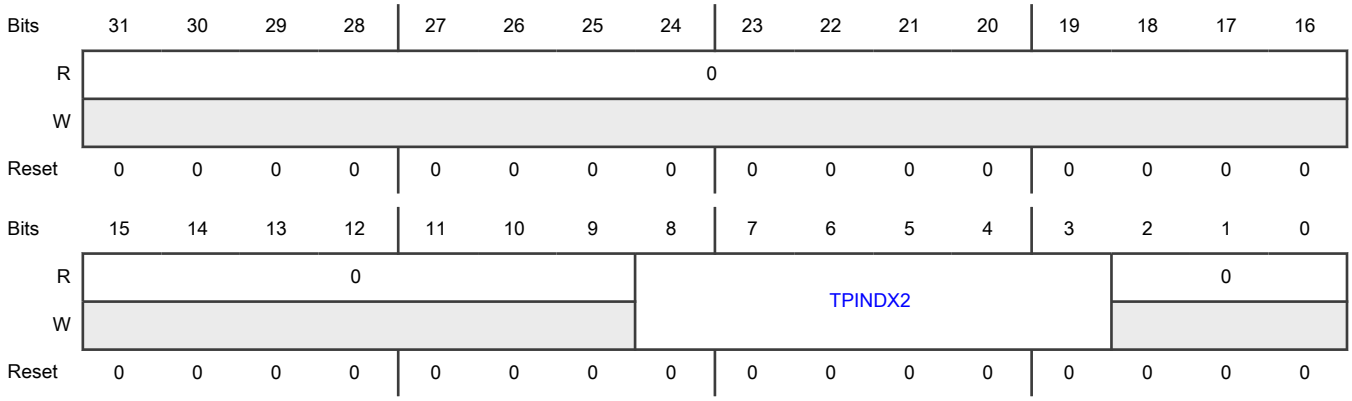
NOTE

The hardware does not provide any protection against illegal programming.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-9 —	Reserved
8-3 TPINDX2	Top index of buffer 2
2-0 —	Reserved

74.10.2.14 DLL Flash Memory A Configuration Register (DLLCRA)

Offset

Register	Offset
DLLCRA	60h

Function

This register configures slave delay chain for flash memory A.

See Delay chain usage for the programming sequence.

NOTE

See the chip data sheet for information on programming register fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	FREQEN	0		Reserved				Reserved				SLV_FINE_OFFSET			
W																
Reset	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SLV_DLY_OFFSET			SLV_DLY_COARSE				Reserved			0	Reserved	SLV_EN	SLV_DLL...	SLV_UPD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 FREQEN	Frequency enable 0b - Selects delay chain for low frequency of operation 1b - Selects delay chain for high frequency of operation
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 SLV_FINE_OFFSET	Fine offset delay elements in incoming DQS This field sets the number of fine offset delay elements up to 16 in incoming DQS, and the default must be 1 element.
15 —	Reserved
14-12 SLV_DLY_OFFSET	T/16 offset delay elements in incoming DQS This field sets the number of T/16 offset delay elements in incoming DQS; default is 0.
11-8 SLV_DLY_COARSE	Delay elements in each delay tap This field sets the number of delay elements in each delay tap. The field is used to overwrite DLL-generated delay values and works when the value of SLV_DLL_BYPASS is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-5 —	Reserved
4 —	Reserved
3 —	Reserved
2 SLV_EN	Slave enable 0b - DLL slave logic remains in reset, and its value should be 0 for at least three flash memory clock cycles for reset. 1b - Enables DQS slave delay chain, and should be 1 before any slave configuration settings take place.
1 SLV_DLL_BYPASS	Slave DLL bypass This field enables selection of the number of delays in each slave delay tap. 0b - Disables manual selection of coarse delays in the slave delay chain. 1b - Enables selection of number of delays in each slave delay tap, based on DLLCRA[SLV_DLY_COARSE].
0 SLV_UPD	Slave update You must program this field only after slave delay chain configuration takes place. 0b - Disables any further update on DQS slave delay chain. 1b - Updates the DQS slave delay chain with either ref-delay or bypass slave delay value, and should be set in the absence of the DQS clock.

74.10.2.15 Serial Flash Memory Address Register (SFAR)

Offset

Register	Offset
SFAR	100h

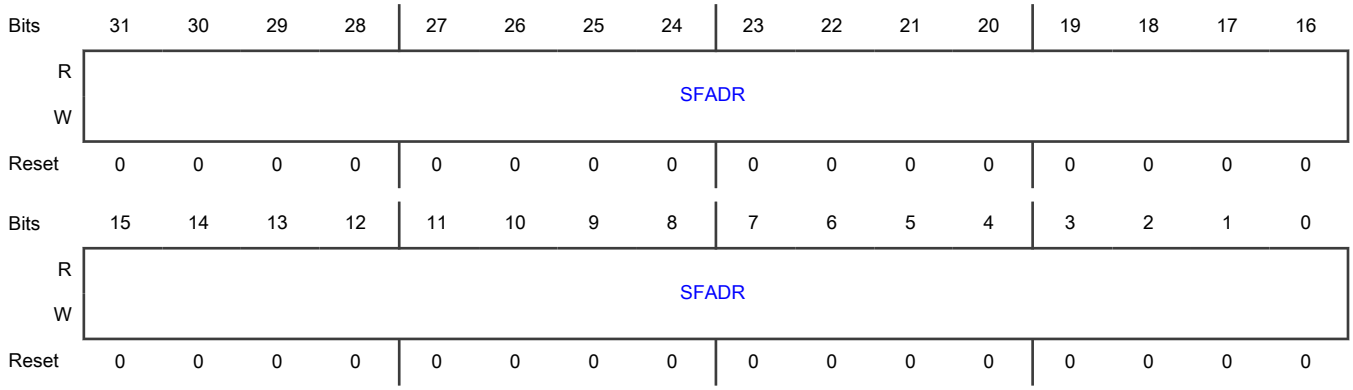
Function

The module automatically translates this address on the memory map to the address on the flash memory. When operating in a 24-bit mode, only bits 23-0 are sent to the flash memory. In the 32-bit mode, bits 27-0 are used with bits 31-28 driven to 0 See Table 569 for the mapping between the access mode and the SFAR content and Normal mode for details on command triggering and command execution. The software must ensure that the serial flash memory address provided in the SFAR register lies in the valid flash memory address range, as defined in Table 569.

Special write-access is permitted if:

- SR[IP_ACC] = 0

Diagram



Fields

Field	Function
31-0 SFADR	Serial flash memory address

74.10.2.16 Sampling Register (SMPR)

Offset

Register	Offset
SMPR	108h

Function

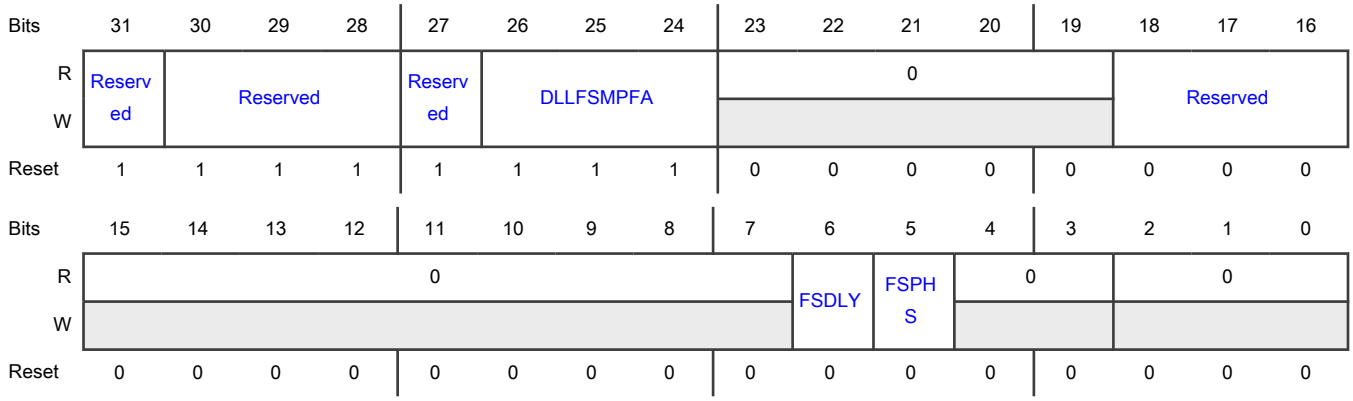
This register allows configuration of how the incoming data from the external serial flash memory devices is sampled in the QuadSPI module.

NOTE

See the chip data sheet for programming the register fields.

Special write-access is permitted in the disabled mode.

Diagram



Fields

Field	Function
31 —	Reserved
30-28 —	Reserved
27 —	Reserved
26-24 DLLFSMPFA	Selects the nth tap provided by slave delay chain for flash memory A The value of *n* can vary from 0 to 7, with each tap delay based on the DLLCRA register.
23-19 —	Reserved
18-16 —	Reserved
15-7 —	Reserved
6 FSDLY	Full speed delay selection for SDR instructions Select the delay with respect to the reference edge for the sample point valid for full speed commands. 0b - One clock cycle delay 1b - Two clock cycles delay
5 FSPHS	Full-speed phase selection for SDR instructions This field selects the edge of the sampling clock valid for full-speed commands. 0b - Select sampling at non-inverted clock

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Select sampling at inverted clock
4-3 —	Reserved
2-0 —	Reserved

74.10.2.17 RX Buffer Status Register (RBSR)

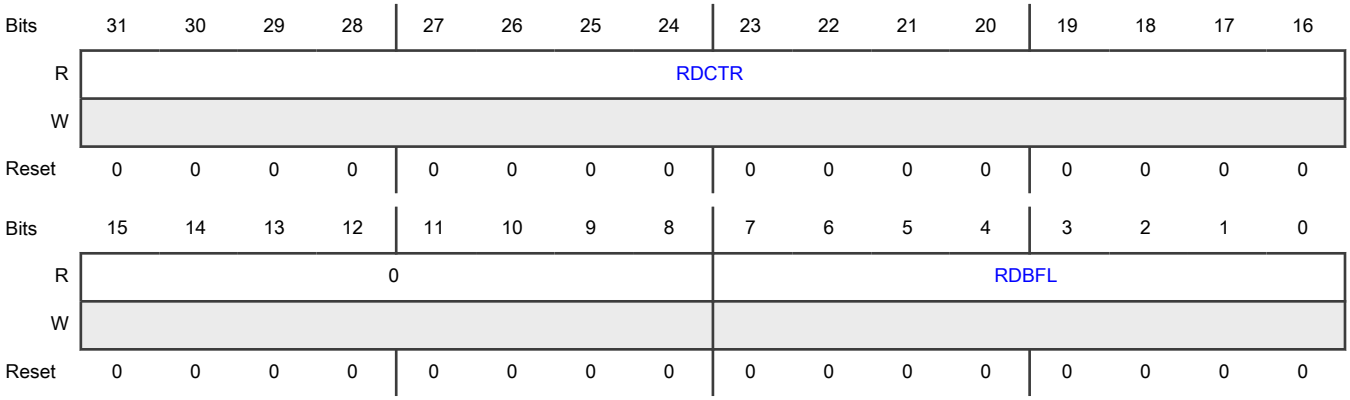
Offset

Register	Offset
RBSR	10Ch

Function

This register contains information related to the receive data buffer.

Diagram



Fields

Field	Function
31-16 RDCTR	Read counter Indicates the number of 4-byte entries removed from the RX buffer. For example, a value of 0x2 indicates that 8 bytes have been removed. It is incremented by the number (RBCT[WMRK] + 1) on RX buffer POP event. The RX buffer can be popped using DMA or FR[RBDF]. The RSER[RBDDE] defines which pop should be pursued. For details, see AHB RX Data Buffer Register (ARDB0 - ARDB127) and Data Transfer from the QuadSPI Module Internal Buffers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 —	Reserved
7-0 RDBFL	RX buffer fill level Indicates the number of 4-byte entries available in the RX buffer. For example, a value of 0x2 indicates 8 bytes are available.

74.10.2.18 RX Buffer Control Register (RBCT)

Offset

Register	Offset
RBCT	110h

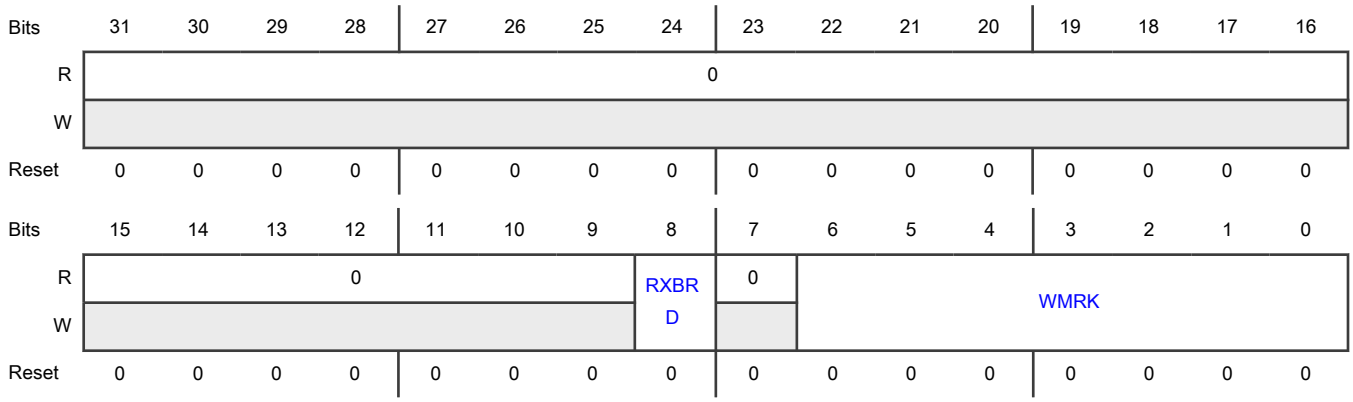
Function

This register contains control data related to the receive data buffer.

Special write-access is permitted if:

- SR[IP_ACC] = 0

Diagram



Fields

Field	Function
31-9 —	Reserved
8	RX buffer readout

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXBRD	This field specifies the access scheme for the RX buffer readout. 0b — RX buffer content is read using the AHB bus registers. See AHB RX Data Buffer Register (ARDB0 - ARDB127). 1b — RX buffer content is read using the IP bus registers. See RX Buffer Data Register (RBDR0 - RBDR63).
7 —	Reserved
6-0 WMRK	RX buffer watermark This field determines when the readout action of the RX buffer is triggered. When the number of valid entries in the RX buffer is equal to or greater than the number given by (WMRK+1), the SR[RXWE] flag is asserted. The value should be entered as the number of 4-byte entries minus 1. For example, a value of 0x0 sets the watermark to 4 bytes, 1 to 8 bytes, 2 to 12 bytes, and so on. For details, see DMA usage.

74.10.2.19 Data Learning Status Flash Memory A Register (DLSR_FA)

Offset

Register	Offset
DLSR_FA	134h

Function

This register shows sampling point selected by data learning algorithm when the value of DLSR_FA[DLPFFA] is 0. Otherwise, it shows the pattern matching outline.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPFFA	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POS_EDGE								NEG_EDGE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLPFFA	Data learning pattern fail This field asserts when data learning fails at flash memory A.
30-16 —	Reserved
15-8 POS_EDGE	DLP positive edge match signature for flash memory A
7-0 NEG_EDGE	DLP negative edge match signature for flash memory A

74.10.2.20 TX Buffer Status Register (TBSR)

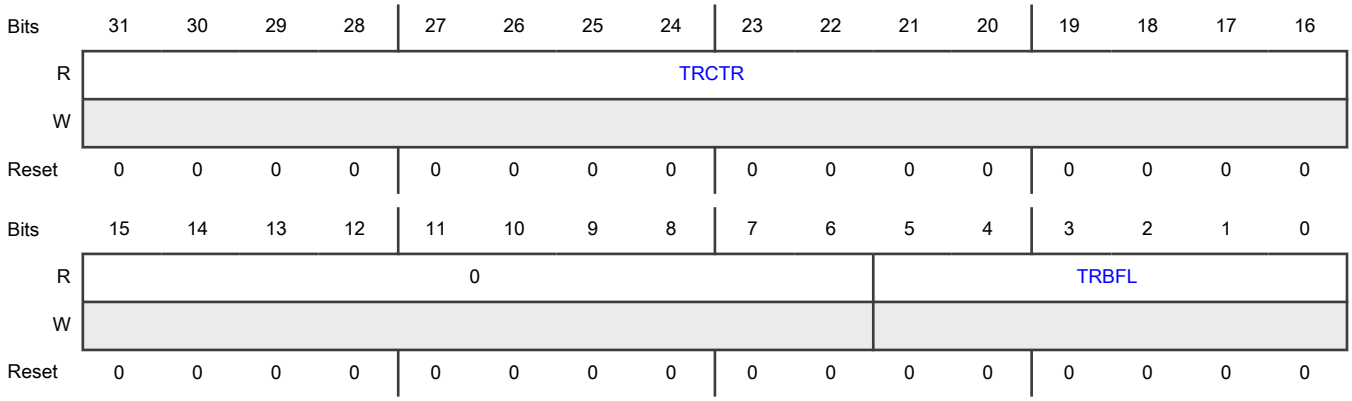
Offset

Register	Offset
TBSR	150h

Function

This register contains information related to the transmit data buffer.

Diagram



Fields

Field	Function
31-16	Transmit counter

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRCTR	This field indicates how many entries of 4 bytes have been written into the TX buffer by host accesses. It is reset to 0 when a 1 is written to MCR[CLR_TXF]. It is incremented on each write access to the TBDR register when another word has been pushed onto the TX buffer. When it is not cleared, the TRCTR field wraps around to 0. See TX Buffer Data Register (TBDR) for details.
15-6 —	Reserved
5-0 TRBFL	TX buffer fill level This field contains the number of entries of 4 bytes each available in the TX buffer for the QuadSPI module to transmit to the serial flash memory device. The value of this field can reach maximum up to the total TX buffer size.

74.10.2.21 TX Buffer Data Register (TBDR)

Offset

Register	Offset
TBDR	154h

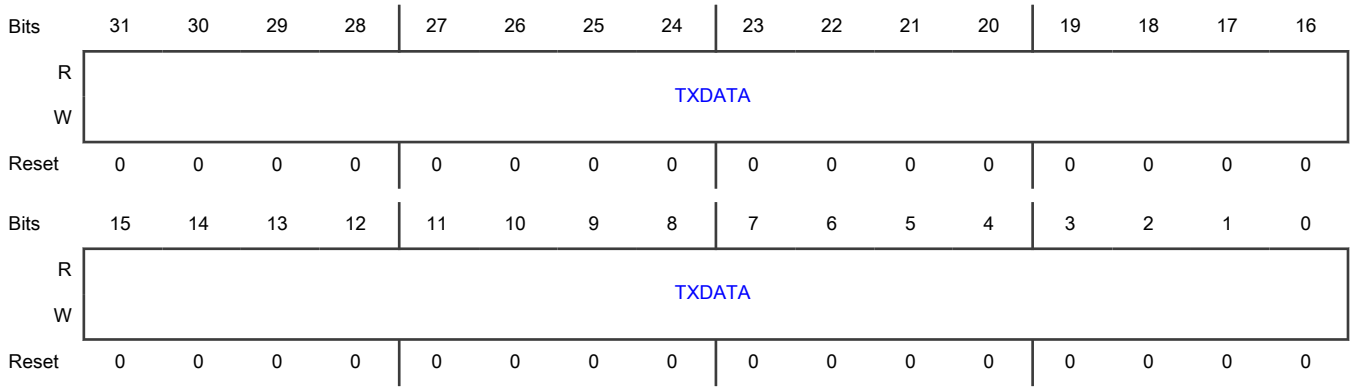
Function

This register provides access to the circular TX buffer of depth 32 , so the total size is 32 * 4 bytes. This buffer provides the data written into it as write data for the page programming commands to the serial flash memory device. See Table 549 for the byte ordering scheme. A write transaction on the flash memory with data size of less than 32 bits leads to the removal of one data entry from the TX buffer. The valid bits are used and the rest of the bits are discarded.

Special write-access is permitted if:

- SR[TXFULL] = 0

Diagram



Fields

Field	Function
31-0 TXDATA	TX data On write access, the data is written to the next available entry of the TX buffer and TBSR[TRBFL] is updated accordingly. On a read access, the last data written to the register is returned.

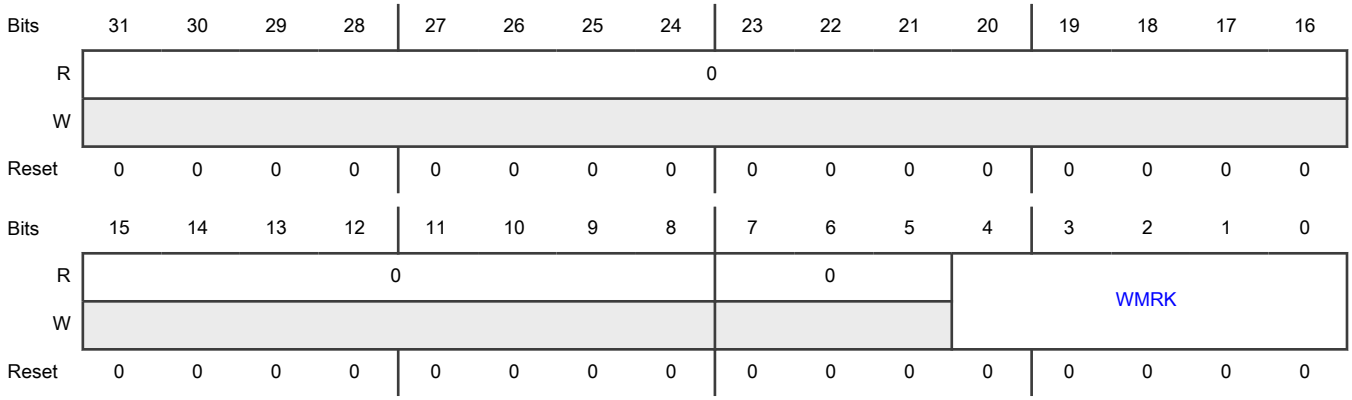
74.10.2.22 TX Buffer Control Register (TBCT)

Offset

Register	Offset
TBCT	158h

Function
This register contains control information for transmit data buffer.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-5 —	Reserved
4-0 WMRK	Watermark for TX buffer Determines the watermark for the TX buffer

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When the number of available space in the TX buffer is greater than or equal to the number provided by WMRK (number of 4-byte entries), SR[TXWA] is asserted. For example, a value of 0x1 sets the watermark to 4 bytes, 0x2 sets it to 8 bytes, 0x3 sets it to 12 bytes, and so on. For details, see DMA usage. WMRK = 0 is invalid.

74.10.2.23 Status Register (SR)

Offset

Register	Offset
SR	15Ch

Function

This register provides all the available status information about SFM command execution and arbitration, the RX buffer, TX buffer, and the AHB buffer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			0	TXFULL	TXDMA	TXWA	TXNE	RXDMA	0			RXFULL	0		RXWE
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	AHB3FUL	AHB2FUL	AHB1FUL	AHB0FUL	AHB3NE	AHB2NE	AHB1NE	AHB0NE	AHBTRN	0	0	0	AHB_ACC	IP_ACC	BUSY
W																
Reset	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 —	Reserved
27	TX buffer full

Table continues on the next page...

Table continued from the previous page...

Field	Function
TXFULL	Asserted when the FIFO level reaches 39 (that is, TX buffer size of 32 + async FIFO size of 7)
26 TXDMA	TX DMA Asserted when the TXFIFO fill via DMA is active and DMA is requested or running
25 TXWA	TX buffer watermark available Asserted when the number of available spaces in the TX buffer is greater than or equal to the value provided by TBCT[WMRK]
24 TXNE	TX buffer not empty Asserted when TX buffer contains data
23 RXDMA	RX buffer DMA Asserted when RX buffer read out via DMA is active; that is, when DMA is requested or running
22-20 —	Reserved
19 RXFULL	RX buffer full Asserted when the RX buffer is full; that is, when RBSR[RDBFL] is equal to 32
18-17 —	Reserved
16 RXWE	RX buffer watermark exceeded Asserted when the number of valid entries in the RX buffer exceeds the number provided in RBCT[WMRK]
15 —	Reserved
14 AHB3FUL	AHB 3 buffer full Asserted when AHB 3 buffer is full
13 AHB2FUL	AHB 2 buffer full Asserted when AHB 2 buffer is full
12 AHB1FUL	AHB 1 buffer full Asserted when the AHB 1 buffer is full
11 AHB0FUL	AHB 0 buffer full Asserted when the AHB 0 buffer is full
10	AHB 3 buffer not empty

Table continues on the next page...

Table continued from the previous page...

Field	Function
AHB3NE	Asserted when the AHB 3 buffer contains data
9 AHB2NE	AHB 2 buffer not empty Asserted when the AHB 2 buffer contains data
8 AHB1NE	AHB 1 buffer not empty Asserted when the AHB 1 buffer contains data
7 AHB0NE	AHB 0 buffer not empty Asserted when the AHB 0 buffer contains data
6 AHBTRN	AHB access transaction pending Asserted when there is a pending request on the AHB interface. See Flash memory mapped AMBA bus.
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 AHB_ACC	AHB read access Asserted when the currently executed transaction is initiated by the AHB bus
1 IP_ACC	IP access Asserted when transaction currently executed is initiated by the IP bus
0 BUSY	Module busy Asserted when module is currently busy handling a transaction to an external flash memory device

74.10.2.24 Flag Register (FR)

Offset

Register	Offset
FR	160h

Function

This register provides all available flags about SFM command execution and arbitration, which may serve as the source for the generation of interrupt service requests. Note that the error flags in this register do not relate directly to the execution of the transaction in the serial flash memory device itself but only to the behavior and conditions visible in the QuadSPI module.

Special write-access is permitted in the enabled mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0		0	TBFF	TBUF	0	0	ILLINE	0		0	0		RBOF	RBDF
W					W1C	W1C			W1C						W1C	W1C
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	AITEF	AIBSEF	ABOF	Reserved	0	0	0	IPAEF	IPIEF	0	0	0			TFF
W		W1C	W1C	W1C	W1C				W1C	W1C						W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-29 —	Reserved
28 —	Reserved
27 TBFF	TX buffer fill flag Before writing to the TX buffer, this field should be cleared. Then, it should be read back. If it is set, the TX buffer can include more data. If the field remains cleared, the TX buffer can be considered as full. See TX buffer operation for details.
26 TBUF	TX buffer underrun flag This field is set if the module tries to pull data when the TX buffer is empty. The IP command leading to the TX buffer underrun is continued (data sent to the serial flash memory device is undefined). Here, a valid underrun means that it should have occurred during the transaction so that few bytes (that is, less than 4 bytes) are left in FIFO and the remaining are filled with "FFFFh". The software should initiate a TX transaction only when 128 bits are written in the TX buffer. This field does not set if transfer is less than 128 bits. The application must clear the TX buffer in response to this event by writing a 1 to MCR[CLR_TXF]. The application must clear the TX buffer in response to this event by writing a 1 to MCR[CLR_TXF].
25 —	Reserved
24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 ILLINE	Illegal instruction error flag This field is set when an illegal instruction is encountered by the controller in any of the sequences. As soon as the field is set, you must assert MCR[SWRSTSD] and MCR[SWRSTHD]. That is, reset the flash memory and AHB domain after reconfiguring the correct sequence instruction. See Table 547 for a list of legal instructions.
22-21 —	Reserved
20 —	Reserved
19-18 —	Reserved
17 RBOF	RX buffer overflow flag This field is set when no more data can be pushed into the RX buffer from the serial flash memory device. The IP command leading to this condition is continued until the number of bytes in IPCR[IDATSZ] are read from the serial flash memory device. The content of the RX buffer remains unchanged.
16 RBDF	RX buffer drain flag This field is set if SR[RXWE] is asserted. Writing 1 to this field triggers one of the following actions: • If the RX buffer has up to RBCT[WMRK] valid entries, then the flag is cleared. • If the RX buffer has more than RBCT[WMRK] valid entries and the RSER[RBDDE] field is not set (flag driven mode), an RX buffer POP event is triggered. The flag remains set if the RX buffer contains more than RBCT[WMRK] valid entries after the RX buffer POP event is complete. The flag is cleared if the RX buffer contains less than or equal to RBCT[WMRK] valid entries after the RX buffer POP event is complete. See the "Receive Buffer Drain Interrupt or DMA Request" section in Normal mode interrupt and DMA requests for details.
15 —	Reserved
14 AITEF	AHB illegal transaction error flag This is set whenever there is no response generated from QuadSPI to AHB bus in case of an illegal transaction and the watchdog timer expires. The timer value is considered as a parameter.
13	AHB illegal burst size error flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
AIBSEF	This is set whenever the total burst size (size x beat) of an AHB transaction is greater than the prefetch data size, which is defined by BUFxCR[ADATSZ] or data size mentioned in the sequence pointed to by the SEQID field in case ADATSZ = 0. See HBURST support with AHB read details on HBURST feature.
12 ABOF	AHB buffer overflow flag This is set when the size of the AHB access exceeds the size of the AHB buffer. This condition can occur only if BUFxCR[ADATSZ] is programmed incorrectly. The AHB command leading to this condition is continued until the number of entries according to BUFxCR[ADATSZ] have been read from the serial flash memory device. The content of the AHB buffer is not changed.
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 IPAEF	IP command trigger during AHB access error flag This is set when the following condition occurs: • A write access occurs to IPCR[SEQID] and the SR[AHB_ACC] field is set. Any command leading to the assertion of the IPAEF field is ignored.
6 IPIEF	IP command trigger could not be executed error flag This is set when the SR[IP_ACC] and SR[AWRACC] fields are set (that is, an IP triggered command is currently executing) and any of the following conditions occurs: • Write access to the IPCR. Any command leading to the assertion of the IPIEF flag is ignored. • Write access to the SFAR • Write access to the RBCT
5 —	Reserved
4 —	Reserved
3-1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 TFF	IP command transaction finished flag This field is set after the QuadSPI module completes a running IP command. If an error occurs, and the related error flags are valid in the same clock cycle, the TFF flag is asserted.

74.10.2.25 Interrupt and DMA Request Select and Enable Register (RSER)

Offset

Register	Offset
RSER	164h

Function

This register provides enables and selectors for the interrupts in the QuadSPI module.

NOTE

Each field of the FR enabled as source for an interrupt prevents the QuadSPI module from entering the Stop mode or Module Disable mode when this flag is set.

Special write-access is permitted in the "Anytime" mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved		0	TBFIE	TBUIE	TBFDE	Reserved	ILLINIE	Reserved	RBDDE	Reserved	0		RBOIE	RBDIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	AITIE	AIBSIE	ABOIE	Reserved	Reserved	Reserved	Reserved	IPAEIE	IPIEIE	0	Reserved	0			TFIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-29 —	Reserved
28 —	Reserved
27 TBFIE	TX buffer fill interrupt enable flag This field indicates the TX buffer fill interrupt enable flag. 0b - No TBFF interrupt is generated. 1b - TBFF interrupt is generated.
26 TBUIE	TX buffer underrun interrupt enable flag This field indicates the TX buffer underrun interrupt enable flag. 0b - No TBUF interrupt is generated 1b - TBUF interrupt is generated
25 TBFDE	TX buffer fill DMA enable Enables generation of DMA requests for TX buffer fill. When the value of this field is 1, DMA requests are generated as long as SR[TXWA] is set. **NOTE** After you write 1 to this field (to enable DMA transfers), writing 0 does not disable DMA transfers. You must perform a software reset for the AHB domain by using MCR[SWRSTHD] to disable DMA transfers. 0b - No DMA request is generated 1b - DMA request is generated
24 —	Reserved
23 ILLINIE	Illegal instruction error interrupt enable Triggered by the ILLINE flag in FR 0b - No ILLINE interrupt is generated. 1b - ILLINE interrupt is generated.
22 —	Reserved
21 RBDDE	RX buffer drain DMA enable This field enables generation of DMA requests for RX buffer drain. When the value of this field is 1, the DMA requests are generated as long as SR[RXWE] is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE After you write 1 to this field (to enable DMA transfers), writing 0 does not disable DMA transfers. You must perform a software reset for the AHB domain by using MCR[SWRSTHD] to disable DMA transfers. 0b - No DMA request is generated. 1b - DMA request is generated.
20 —	Reserved
19-18 —	Reserved
17 RBOIE	RX buffer overflow interrupt enable This field indicates the RX buffer overflow interrupt enable flag. 0b - No RBOF interrupt is generated. 1b - RBOF interrupt is generated.
16 RBDIE	RX buffer drain interrupt enable This field enables generation of IRQ requests for RX buffer drain. When the value of this field is 1, the interrupt is asserted as long as SR[RBDF] is set. 0b - No RBDF interrupt is generated. 1b - RBDF Interrupt is generated.
15 —	Reserved
14 AITIE	AHB illegal transaction interrupt enable flag This field indicates the AHB illegal transaction interrupt enable flag. 0b - No AITEF interrupt is generated. 1b - AITEF interrupt is generated.
13 AIBSIE	AHB illegal burst size interrupt enable flag This field indicates the AHB illegal burst size interrupt enable flag. 0b - No AIBSEF interrupt is generated. 1b - AIBSEF interrupt is generated.
12 ABOIE	AHB buffer overflow interrupt enable flag This field indicates the AHB buffer overflow interrupt enable flag.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No ABOF interrupt is generated. 1b - ABOF interrupt is generated.
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 IPAEIE	IP command trigger during AHB read access error interrupt enable flag This field indicates IP command trigger during AHB read access error interrupt enable flag. 0b - No IPAEF interrupt is generated 1b - IPAEF interrupt is generated
6 IPIEIE	IP command trigger during IP access error interrupt enable flag This field indicates IP command trigger during IP access error interrupt enable flag. 0b - No IPIEF interrupt is generated 1b - IPIEF interrupt is generated
5 —	Reserved
4 —	Reserved
3-1 —	Reserved
0 TFIE	Transaction finished interrupt enable flag This field indicates the transaction finished interrupt enable flag. 0b - No TFF interrupt is generated. 1b - TFF interrupt is generated.

74.10.2.26 Sequence Pointer Clear Register (SPTRCLR)

Offset

Register	Offset
SPTRCLR	16Ch

Function

This register provides fields to reset the IP and buffer sequence pointers. The sequence pointer contains the index of instructions within the LUT entry that is to be executed next. For example, if the LUT entry ends on a JMP_ON_CS value of 2, the index is stored as 2.

The software should reset the sequence pointers whenever the sequence ID is changed by updating the SEQID field in the IPCR or BFGENCR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						0		0				0		Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							0	0							0
W								IPPTRC								BFPTRC
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25-24 —	Reserved
23-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-9 —	Reserved
8 IPPTRC	IP pointer clear This is a self-clearing field. 1b - Clears the sequence pointer for IP accesses as defined in IPCR.
7-1 —	Reserved
0 BFPTRC	Buffer pointer clear This is a self-clearing field. 1b - Clears the sequence pointer for AHB read accesses as defined in BFGENCR.

74.10.2.27 Serial Flash Memory A1 Top Address Register (SFA1AD)

Offset

Register	Offset
SFA1AD	180h

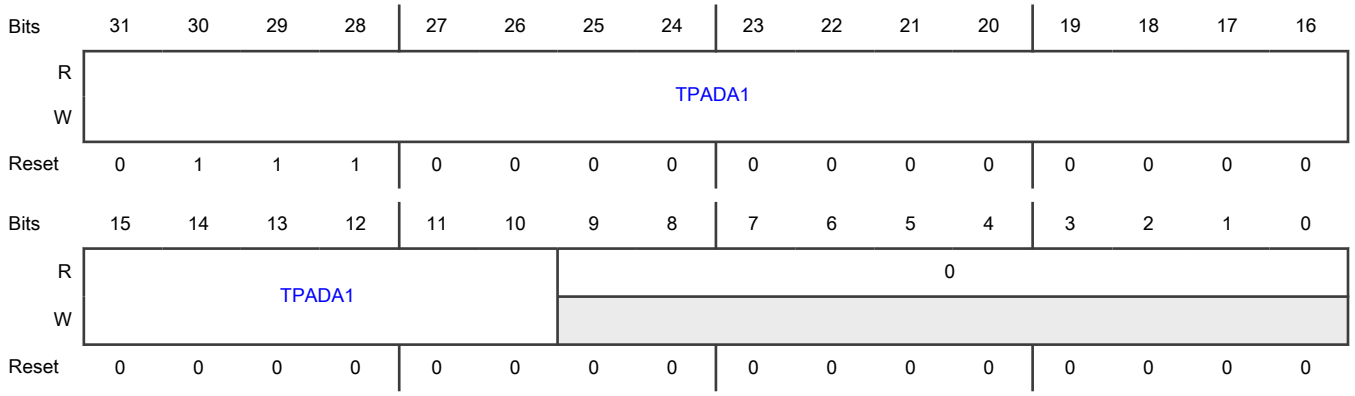
Function

This register provides the address mapping for serial flash memory A1. The difference between SFA1AD[TPADA1] and AMBA_BASE defines the size of the memory map for serial flash memory A1.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-10 TPADA1	Top address for serial flash memory A1 In effect, TPADxx is the first location of the next memory.
9-0 —	Reserved

74.10.2.28 Serial Flash Memory A2 Top Address Register (SFA2AD)

Offset

Register	Offset
SFA2AD	184h

Function

This register provides the address mapping for serial flash memory A2. The difference between SFA2AD[TPADA2] and SFA1AD[TPADA1] defines the size of the memory map for serial flash memory A2.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TPADA2															
Reset	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TPADA2						0									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 TPADA2	Top address for serial flash memory A2 In effect, TPxxAD is the first location of the next memory.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-0 —	Reserved

74.10.2.29 Serial Flash Memory B1 Top Address Register (SFB1AD)

Offset

Register	Offset
SFB1AD	188h

Function

This register provides the address mapping for serial flash memory B1. The difference between SFB1AD[TPADB1] and SFA2AD[TPADA2] defines the size of the memory map for serial flash memory B1.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TPADB1															
Reset	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TPADB1						0									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 TPADB1	Top address for serial flash memory B1. In effect, TPxxAD is the first location of the next memory.
9-0 —	Reserved

74.10.2.30 Serial Flash Memory B2 Top Address Register (SFB2AD)

Offset

Register	Offset
SFB2AD	18Ch

Function

This register provides the address mapping for serial flash memory B2. The difference between SFB2AD[TPADB2] and SFB1AD[TPADB1] defines the size of the memory map for serial flash memory B2.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TPADB2															
Reset	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TPADB2						0									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 TPADB2	Top address for serial flash memory B2. In effect, TPxxAD is the first location of the next memory.
9-0 —	Reserved

74.10.2.31 RX Buffer Data Register (RBDR0 - RBDR63)

Offset

For a = 0 to 63:

Register	Offset
RBDRa	200h + (a × 4h)

Function

These registers provide access to individual entries in the RX buffer. See Table 549 for the byte ordering scheme.

RBDR0 corresponds to the actual position of the read pointer within the RX buffer. The number of valid entries available depends on the number of RX buffer entries implemented and on the number of valid buffer entries available in the RX buffer.

Example 1 - RX buffer filled completely with 128 words: In this case, the address range for valid read access extends from RBDR0 to RBDR63.

Example 2 - RX buffer filled with five valid words: RX buffer fill level of RBSR[RDBFL] is 5. In this case, access to RBDR4 provides the last valid entry.

Any access beyond the range of valid RX buffer entries provides undefined results.

NOTE

To access data beyond RBDR[63], you must pop the data by using FR[RBDF]. See here for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXDATA	RX data This field contains the data associated with the related RX buffer entry. For data format and byte ordering, see Byte ordering of serial flash memory read data.

74.10.2.32 LUT Key Register (LUTKEY)

Offset

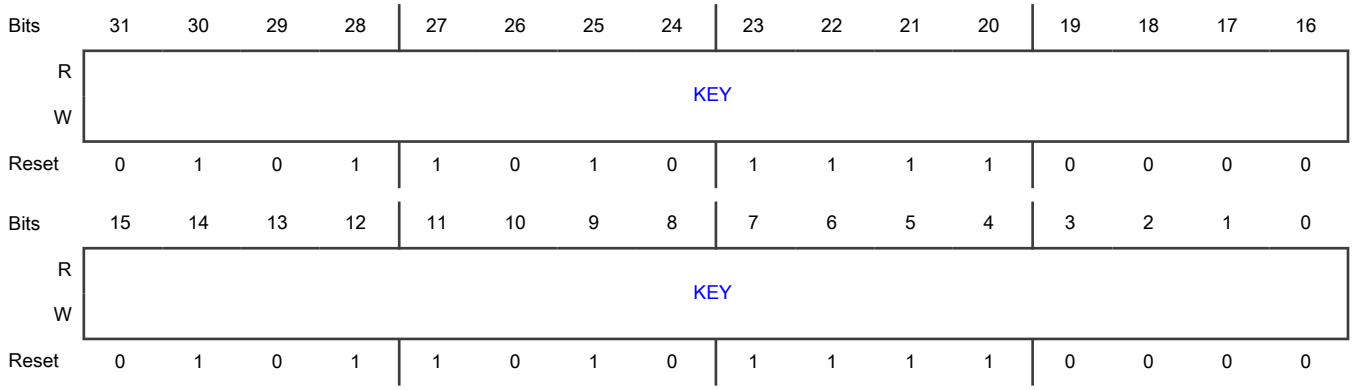
Register	Offset
LUTKEY	300h

Function

This register contains the key to lock and unlock the LUT. See LUT for details.

Special write-access is permitted in the "Anytime" mode.

Diagram



Fields

Field	Function
31-0 KEY	Key to lock or unlock the LUT The key is 0x5AF05AF0 and the read value is always 0x5AF05AF0.

74.10.2.33 LUT Lock Configuration Register (LCKCR)

Offset

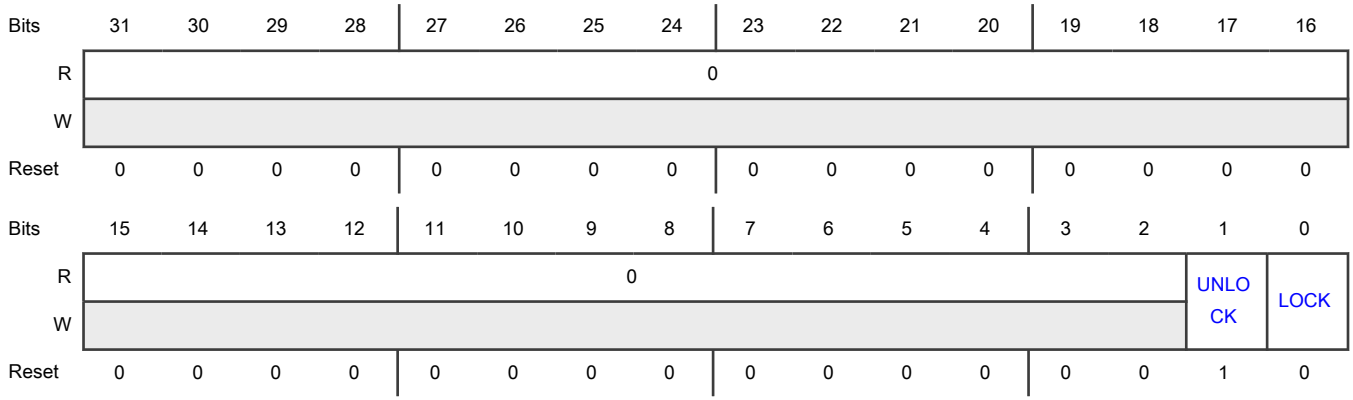
Register	Offset
LCKCR	304h

Function

This register is used along with the LUTKEY register to lock or unlock the LUT. This register should be written immediately after the LUTKEY register for the lock or unlock operation to be successful. See LUT for details. Setting both the LOCK and UNLOCK bits as "00" or "11" is not allowed.

Special write access is permitted after writing the LUT key register.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 UNLOCK	Unlock LUT Unlocks the LUT when the following two conditions are met: • This register is written just after the LUT Key Register (LUTKEY). • The LUT key register was written with the 0x5AF05AF0 key.
0 LOCK	Lock LUT Locks the LUT when the following conditions are met: • This register is written just after the LUT Key Register (LUTKEY). • The LUT key register is written with the 0x5AF05AF0 key.

74.10.2.34 LUT Register (LUT0)

Offset

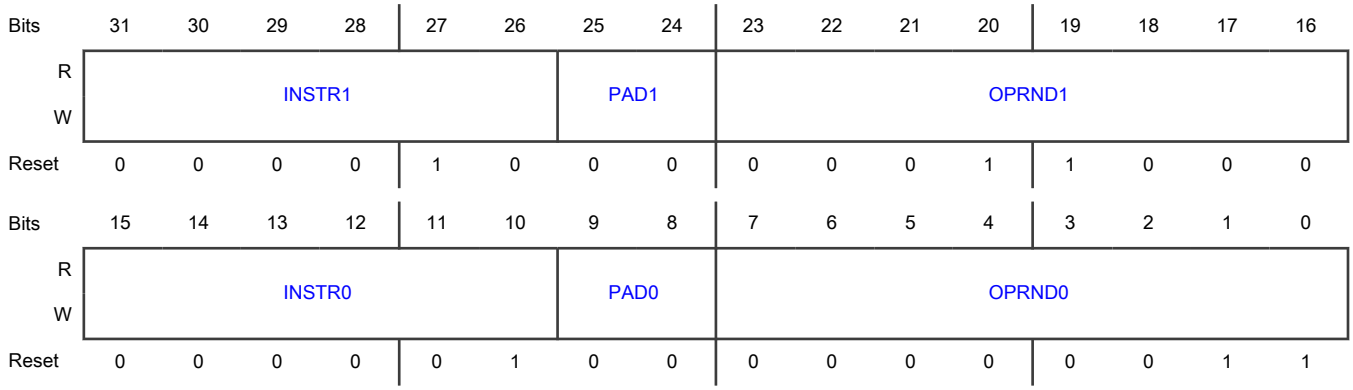
Register	Offset
LUT0	310h

Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 20 LUT registers. These 20 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[15] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT19. A maximum of 4 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

Diagram



Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
23-16 OPRND1	Operand for INSTR1
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
7-0 OPRND0	Operand for INSTR0

74.10.2.35 LUT Register (LUT1)

Offset

Register	Offset
LUT1	314h

Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 20 LUT registers. These 20 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[15] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT19. A maximum of 4 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INSTR1						PAD1		OPRND1							
Reset	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INSTR0						PAD0		OPRND0							
Reset	0	0	0	1	1	1	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
23-16	Operand for INSTR1

Table continues on the next page...

Table continued from the previous page...

Field	Function
OPRND1	
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
7-0 OPRND0	Operand for INSTR0

74.10.2.36 LUT Register (LUT2 - LUT19)

Offset

For a = 2 to 19:

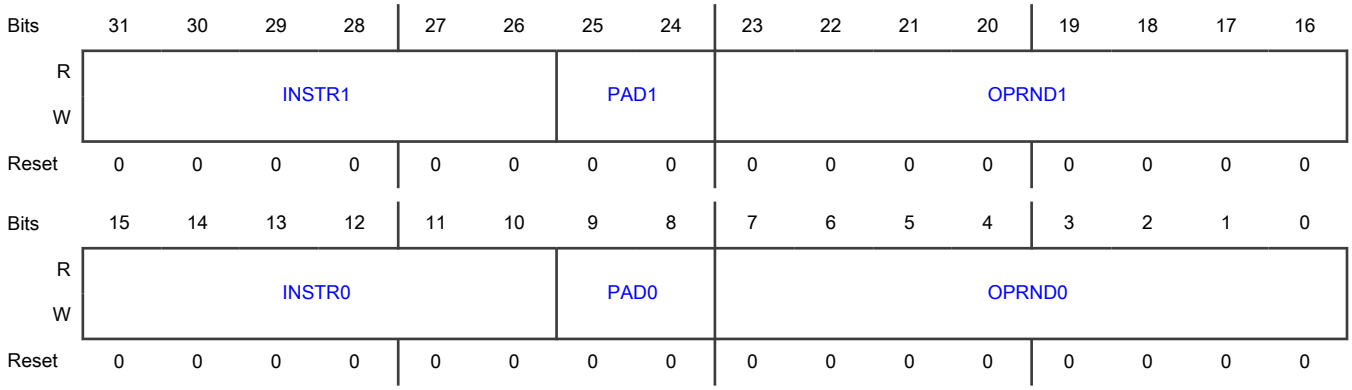
Register	Offset
LUTa	310h + (a × 4h)

Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 20 LUT registers. These 20 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[15] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT19. A maximum of 4 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

Diagram



Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
23-16 OPRND1	Operand for INSTR1
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
7-0 OPRND0	Operand for INSTR0

74.10.3 Serial flash memory address assignment

The serial flash memory address assignment can be modified by writing into Serial Flash Memory A1 Top Address Register (SFA1AD) and Serial Flash Memory A2 Top Address Register (SFA2AD) for device A

Table 569. Serial flash memory address assignment

Parameter	Function	Access mode
QuadSPI_AMBA_BASE ((31:10) - 22 bits)	QuadSPI AHB base address First address of the serial flash memory device as presented to the QuadSPI controller. This might be the base of the serial flash memory in the system address map or it may be a remapping, for instance to 0h, performed by the system. (See the system address map file attached to this document)	
QuadSPI_ARDB_BASE	First address of the QuadSPI Rx buffer on system memory map	
TOP_ADDR_MEMA1(TPADA1)	Top address for the external flash memory A1 (the first of the two independent flash memories sharing the IOFA)	Any access to the address space between TOP_ADDR_MEMA1 and QuadSPI_AMBA_BASE is routed to serial flash memory A1.
TOP_ADDR_MEMA2(TPADA2)	Top address for the external flash memory A2 (the second of the two independent flash memories sharing the IOFA)	Any access to the address space between TOP_ADDR_MEMA2 and TOP_ADDR_MEMA1 is routed to serial flash memory A2.

74.11 Flash memory mapped AMBA bus

QuadSPI_AMBA_BASE defines the address to be used as the start address of the serial flash memory device, as defined by the system memory map. Note that this may be a remapping of the physical address of the serial flash memory in the system. See the system address map for details.

Table 570. QuadSPI AMBA bus memory map

Address	Register name
QuadSPI_AMBA_BASE to (TOP_ADDR_MEMA2 - 1h)	• See Memory-mapped serial flash memory data—individual flash memory mode on flash memory A. • For information about byte ordering, see Table 549 and Table 550.
QuadSPI_ARDB_BASE to... (32 * 4 Byte) QuadSPI_ARDB_BASE + 1FFh	• See AHB RX Data Buffer Register (ARDB0 - ARDB127). • For information about the byte ordering, see Table 549.

NOTE

Any read access to non-implemented addresses provides undefined results.

In individual flash memory modes, the 3/4 address bytes (as programmed in the instruction/operand in the sequence) available for the flash memory address are determined by SFADR[23:0] or SFADR[31:0] as provided in the table shown above.

74.11.1 AHB bus access read considerations

Note that all logic in the QuadSPI module implementing the AHB bus access is designed to read the content of an external serial flash memory device. Therefore, the following restrictions apply to the QuadSPI module with respect to accesses to the AHB bus:

- Any AHB command resulting in the assertion of the FR[ABSEF] flag is answered with the ERROR condition according to the AMBA_AHB specification. The resulting AHB command is ignored.
- AHB bus read access types—NONSEQ and BUSY—are fully supported.
- AHB read access type—SEQ—is treated in the same way as NONSEQ. See Flash memory mapped AMBA bus for details.
- Early burst termination is not supported for AHB transactions.

74.11.2 Memory-mapped serial flash memory data—individual flash memory mode on flash memory A

Starting with address QuadSPI_AMBA_BASE, the content of the first external serial flash memory device is mapped into the address space of the device containing the QuadSPI module. Serial flash memory byte address 0h corresponds to bus address, QuadSPI_AMBA_BASE, in an increasing order. . See the following table for the address mapping. The byte ordering for 32-bit access is provided in Table 549 and for 64-bit read access, the byte ordering is provided in Table 550.

Table 571. Memory-mapped individual flash memory mode—flash memory A address scheme

Memory-mapped address 32-bit access	Memory-mapped address 64-bit access	Serial flash memory byte address	Flash memory device
QuadSPI_AMBA_BASE + 0h	QuadSPI_AMBA_BASE + 0h	0h to 3h	A1
QuadSPI_AMBA_BASE + 4h		4h to 7h	
...	...	...	
TOP_ADDR_MEMA1 - 8h	TOP_ADDR_MEMA1 - 8h	(TOP_ADDR_MEMA1- 8h) to (TOP_ADDR_MEMA1 - 0h4 -0h1)	
TOP_ADDR_MEMA1 - 4h		(TOP_ADDR_MEMA1 - 4h) to (TOP_ADDR_MEMA1 - 0h1)	
TOP_ADDR_MEMA1 + 4h	TOP_ADDR_MEMA1 + 0h	0h to 3h	A2
TOP_ADDR_MEMA1 + 0h4		4h to 7h	
...	...	...	
TOP_ADDR_MEMA2 - 8h	TOP_ADDR_MEMA2 - 8h	(TOP_ADDR_MEMA2 - 8h) to (TOP_ADDR_MEMA2 - 4h - 1h)	
TOP_ADDR_MEMA2 - 4h		(TOP_ADDR_MEMA2 - 4h) to (TOP_ADDR_MEMA2 - 1h)	

The available address range depends on the size of the external serial flash memory device. Any access beyond the size of the external serial flash memory provides undefined results.

For details concerning the read process, see Flash memory read.

74.11.3 ARDB register descriptions

NOTE

See the system memory map in this document for the base address of the QuadSPI AHB RX data buffer.

74.11.3.1 ARDB memory map

QuadSPI_ARDB base address: 6800_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 1FCh	AHB RX Data Buffer Register (ARDB0 - ARDB127)	32	R	0000_0000h

74.11.3.2 AHB RX Data Buffer Register (ARDB0 - ARDB127)

Offset

For a = 0 to 127:

Register	Offset
ARDBa	0h + (a × 4h)

Function

This register is used to read the buffer content of the RX buffer from successive addresses. ARDB0 corresponds to the RX buffer register entry corresponding to the current value of the read pointer in an increasing order.

The increment of the read pointer depends on the access scheme (DMA or flag-driven). See Flash memory read, RX buffer, data read through register interface, and AHB read for the description of successive accesses to the RX buffer content. See Byte Ordering of Serial Flash Memory Read Data for the byte ordering scheme.

Valid address range accessible in the ARDB*n* range depends on the number of RX buffer entries implemented and on the number of valid buffer entries available in the RX buffer.

- Example 1 - RX buffer filled completely with 128 words: In this case, the address range for valid read access extends from ARDB0 to ARDB127.
- Example 2 - RX buffer filled with five valid words; RX buffer fill level RBSR[RDBFL] is 5. In this case, an access to ARDB4 provides the last valid entry.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ARXD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ARXD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ARXD	ARDB provided RX buffer data Byte order (endianness) is identical to the RX buffer data registers.

74.12 Glossary

AHB	Advanced high-performance bus, a version of AMBA
AMBA	Advanced microcontroller bus architecture
APB	Advanced peripheral bus
BE	Big endian byte ordering
CRS	Center aligned read strobe
CS	Chip select
FRAD	Flash region access descriptor
I/O	Input output, I/O lines are also referred to as pads in this chapter
IFM	Individual flash memory mode
LE	Little endian
MDAD	Master domain access descriptors
MGID	Master-Group identifier
PCS	Peripheral chip select
SCK	Serial communications clock
SFM	Serial flash memory

Chapter 75
Debug Subsystem

75.1 Introduction

This chapter discusses the debug and trace architecture of the chip that is based on the specifications provided in the *Arm® CoreSight™ SoC-400 Technical Reference Manual*. See References for a link to this document and to other related documentation available on the Arm website.

The chip architecture includes debug and trace modules. See Features for details on the debug and trace features that Cortex-M7 core clusters support.

The debug components that the Cortex-M7 core supports are accessible via the Arm DAP controller-based architecture. The DAP controller works in parallel with the system JTAGC. Both these controllers share the JTAG port and JTAG instruction set.

The system components are similar for different chips in the same family and include the primary core debug interfaces, the chip-level debug interfaces, and chip-level trace interfaces. The accelerator components include debug and trace circuits. These circuits are necessary for any application-specific accelerators required for the different application spaces that a chip supports. They vary from one chip to another and could include only trace components.

75.2 Interfaces supported in MWCT2S family

Table 572. Interfaces supported in MWCT2S family

MWCT2D17S	MWCT2D16S	MWCT2016S / MWCT2015S / MWCT2014S
Cortex-M7_0 (CTI, DWT, BPU, ETM, ITM)	Cortex-M7_0 (CTI, DWT, BPU, ITM)	Cortex-M7_0 (CTI, DWT, BPU, ITM)
Cortex-M7_1 (CTI, DWT, BPU, ETM, ITM)	Cortex-M7_1 (CTI, DWT, BPU, ITM)	-
-	-	-
Cortex-M0+	Cortex-M0+	Cortex-M0+
HTM (CTI+DMA/EMAC TRACE)	-	-
SWO	SWO	SWO
TPIU	-	-
ETF (4)	-	-
FUNNELs (3)	-	-
CTI (4)	CTI (3)	CTI (2)
CTM (1)	CTM (1)	CTM (1)
Timestamp	Timestamp	Timestamp
MDM_AP	MDM_AP	MDM_AP
SDA_AP	SDA_AP	SDA_AP

75.3 Block diagram

This figure illustrates the DAP architecture of the MWCT2xxxS family.

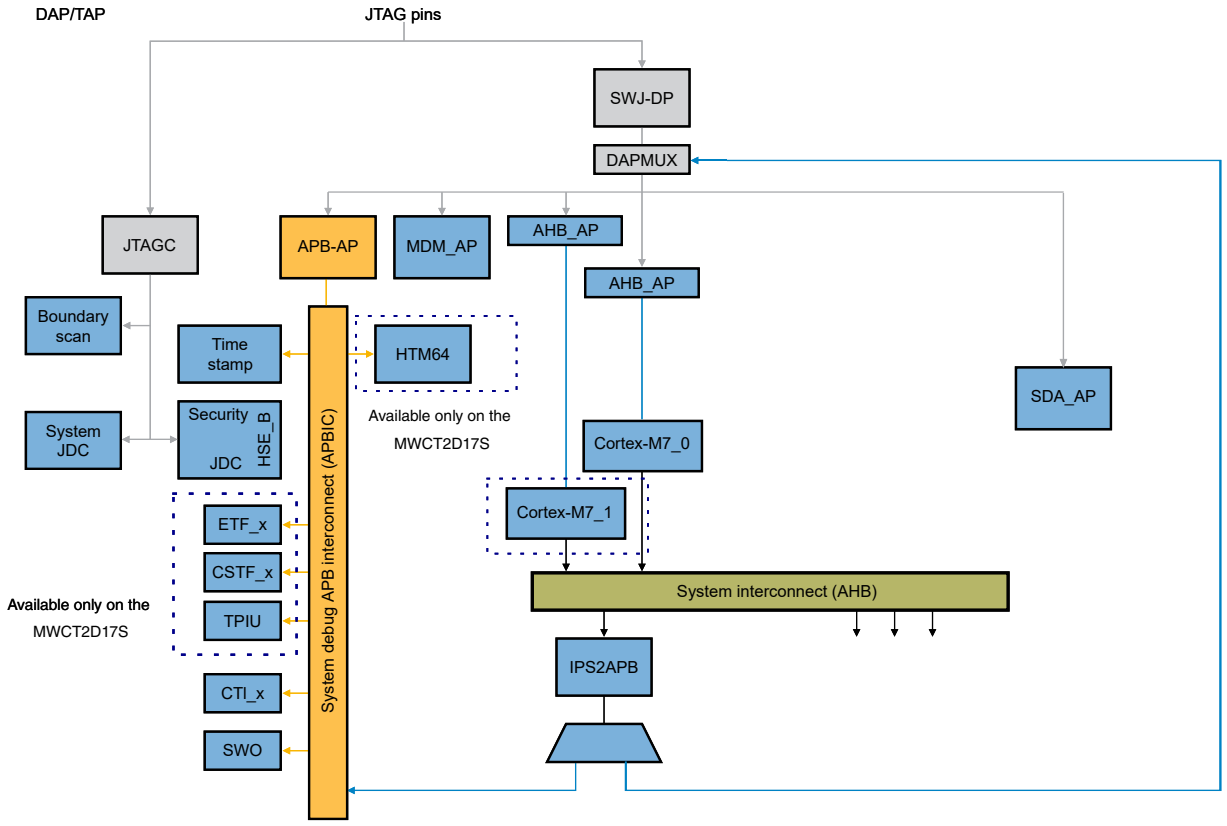


Figure 439. DAP architecture

NOTE

Debug components other than the Cortex-M0+ core are accessible through software path (CORE > AXBS > AIPS > DAPMUX > APs), that is, by directly specifying the memory mapped addresses without doing challenge response.

75.4 Features

This chip includes these features that support the functions listed under each feature type:

- Debug control
  - Is implemented via IEEE 1149.1-compliant JTAG
- Test control
  - Is implemented via IEEE 1149.1-compliant JTAG
  - Uses IEEE 1149.6 extension to IEEE 1149.1
- Trace interface
  - Includes four high-speed data pads and one clock pad up to 125 MHz (125 MB/s throughput per pad)
  - Includes 16 low-speed data pads and one clock pad @25 MHz (100 MB/s throughput per pad)
- Debug security
  - Includes a DAP/TAP interface that controls all debug features and is gated by Cortex M0+ JDC module
  - Includes security modes as described in the "Security" chapter
- Debugging and run control

- Is implemented through stopping points, starting points, breakpoints, and watchpoints
- Each Cortex-M7 core supports eight instruction comparators and a watchpoint unit having four watchpoints

- Trace source
  - Includes Cortex-M7:
    - ETM that provides instruction and data trace
    - ITM that provides DWT, time-stamping, and diagnostic information
- Cross-triggering support
  - Controls run-control options of cores based on other cores
  - Provides a matrix having connections to:
    - HTM via system CTI
    - All Cortex-M7 CPUs
- Ability to view and modify all memory-mapped areas that are not otherwise blocked—includes a system bus debug access port
- Performance monitoring of cores—implements a performance monitoring unit (PMU) on each Cortex core
- Safety
  - Supports monitoring of debug signals to avoid common mode faults
  - Allows checking of erroneous activation of debugging, especially if intrusive (for example, a CPU entering Debug state)
- Timestamps
  - Generates a timestamp bus for distribution to the trace sources
  - Includes a 48-bit binary timestamp bus
  - Clocks timestamp generator by a frequency given in Table 580

75.5 Debug

75.5.1 TAP connectivity

This figure shows a detailed view of TAP connectivity. JTAG select, SWJ-DP, and JTAG-DP are parts of DAP.

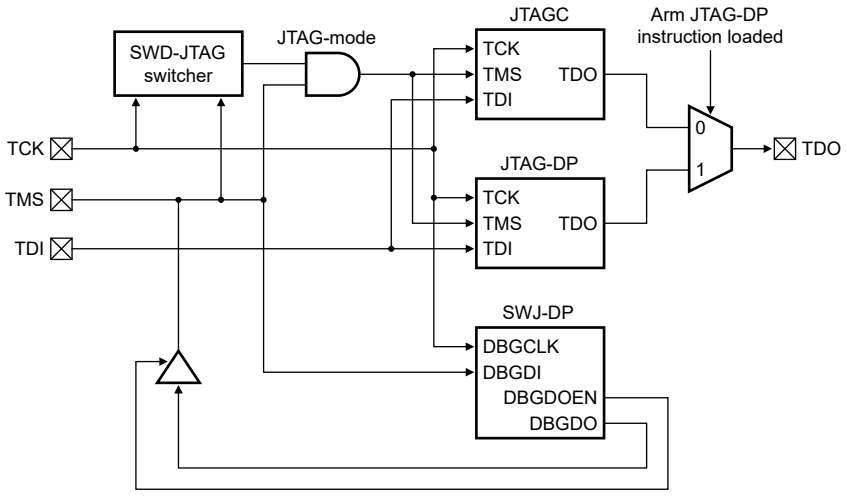


Figure 440. TAP connectivity

The debug port comes out of reset in a standard JTAG mode. It switches to another mode by using the change sequences. After the mode changes, unused debug pins can be reassigned to any of their alternative muxed functions.

JTAG-DP and JTAGC are connected in an overlay scheme and both have an Instruction Register (IR) length of 8 bits.

75.5.2 DAP TAP

DAP is an Arm component that provides multiple-master driving ports. A single external interface port accesses and controls these ports to provide system-wide debug.

The DAP Instruction Register (DAP IR) overlays with the system JTAG Instruction Register (JTAGC IR). Table 573 presents DAP instructions. In addition to the four codes listed in the table, DAP uses BYPASS, which is identical to JTAGC BYPASS and is therefore not shown in the table.

Table 573. DAP IR codes

Code	DAP IR
1111_1000b	ABORT
1111_1010b	DPACC
1111_1011b	APACC
1111_1110b	IDCODE

DAP offers an AHB master interface to access system buses. It also exports the internal DAP bus to extend the access ports. For more information on DAP TAP, see the *Arm Debug Interface Architecture Specification* document available in References.

In this chip:

- The AHB slave ports of all Cortex-M*x* cores provide debugger access to all memory units and registers in the system.
- The new pass-through approach allows the debugger to see a cache coherent view of the memory map for that core.
- The debugger must access the corresponding AHB_AP port to access the AHB-S port of that subsystem.
- XRDC controls system access.
- A core can access the debug components of another core through a DAPMUX to the bus interconnect.
- The exported DAP bus hosts AHB_AP and MDM_AP.
  - MDM_AP hosts system-level JTAG status and control registers (see MDM_AP register descriptions). These registers can be used for cross triggering, synchronized debug, and other miscellaneous control and status functions.
  - APB_AP uses an APSEL value of 1h to access any APB-mapped debug modules.

Table 577 describes the access addresses of APB-mapped debug modules. The value of the DAP select signal in the APSEL field of SWJ-DP's Select register selects the access ports in the DAP. Table 574 shows APSEL decoding.

Table 574. DAP master address mapping

DAP master #	DAP component	DAPBUS base address (access from debugger) [1]	System memory map address (access from cores)		Selected port or master	Applicability
			Page number	Base address		
0h	Reserved	0000_0000h	0	4025_0000h	Reserved	-

Table continues on the next page...

Table 574. DAP master address mapping (continued)

DAP master #	DAP component	DAPBUS base address (access from debugger) [1]	System memory map address (access from cores)		Selected port or master	Applicability
			Page number	Base address		
1h	APB_AP to all APB-mapped debug modules (for example, ETFs, HTM, and trace funnels)	0100_0000h	0	4025_0100h	APB_AP to all APB-mapped debug modules (for example, ETFs, HTM, CSTF, and so on)	MWCT2D17S (for ETF, HTM, and CSTF see Table 572)
2h	Reserved	0200_0000h	0	4025_0200h	Reserved	-
4h	AHB_AP to Cortex-M7_0 debug modules and chip system memory map	0400_0000h	0	4025_0400h	AHB_AP to Cortex-M7_0 debug modules and chip system memory map	All
5h	AHB_AP to Cortex-M7_1 debug modules and chip system memory map	0500_0000h	0	4025_0500h	AHB_AP to Cortex-M7_1 debug modules and chip system memory map	MWCT2D17S, MWCT2x16S
6h	MDM_AP	0600_0000h	0	4025_0600h	MDM_AP	All
7h	SDA_AP	0700_0000h	0	4025_4700h	SDA_AP used for challenge-response (CR) in SWJ-DP mode	All
9h—FFh	Reserved (default AP response)	—	—	—	—	—

1. For example, 4025_XXYYh address is provided when accessing access ports (APs) via the debugger. Here, XX shows the AP select number and YY shows the address to be accessed. If you access MDM_AP via the cores on 4h, you must provide the address 4025_0604h

NOTE

Accessing SDA_AP simultaneously from the core and the debugger is prohibited. If you try to do so, you may receive unpredictable responses to core-initiated transactions (for example, incorrect data read, a failed attempt to write to the register, or a transfer error). Debugger-initiated transactions proceed correctly.

75.5.3 System JTAGC

JTAGC connects in parallel with the TAP controller (JTAG-DP of DAP), which has an IR length of 8 bits. The JTAGC IR codes overlay the ones of the DAP controller. DAP uses four instructions and JTAGC uses the remaining ones. The TAP outputs (TPOs) are multiplexed based on the selected IR code. This chip is fully JTAG-compliant and appears as a single TAP to the JTAG chain.

This table shows the JTAGC IR codes. The instructions that are used by Arm DAP TAP are shown in Table 573.

Table 575. JTAG instructions for JTAGC

Code	JTAGC IR
0000_0000b	IDCODE
0000_0001b	Reserved
0000_0010b	SAMPLE/PRELOAD
0000_0011b	SAMPLE
0000_0100b	EXTEST
0000_0101b	HI-Z
0000_0110b	Reserved
0000_0111b	Reserved
0000_1000b—0000_1001b	Reserved
0000_1010b—0000_1011b	Reserved
0000_1100b	CLAMP
0000_1101b	ENABLE_SOC_DATA1
0000_1110b	Reserved
0000_1111b	Reserved
1000_0000b	Reserved
1000_0001b	Reserved
1000_0010b	Reserved
1000_0011b	Reserved
1000_0100b	Reserved
1000_0101b	Reserved
1000_0110b—1000_0111b	Reserved
1000_1000b	Reserved
1000_1001b	Reserved
1000_1010b—1000_1111b	Reserved
1001_0000b	Security JDC
1001_0001b	System JDC
1001_0010b—1001_0111b	Reserved for other system auxiliary clients

Table continues on the next page...

Table 575. JTAG instructions for JTAGC (continued)

Code	JTAGC IR
1001_1000b—1001_1011b	Reserved
1001_1100b—1011_1111b	Reserved for other system auxiliary clients
1100_0000b	Reserved
1100_0001b	Reserved
1100_0010b—1110_1111b	Reserved
1111_1000b	ABORT (Arm)
1111_1001b	Reserved
1111_1010b	DPACC (Arm)
1111_1011b	APACC (Arm)
1111_1100b	Reserved
1111_1101b	Reserved
1111_1110b	IDCODE (Arm)
1111_1111b	BYPASS

75.5.3.1 Chip JTAG/Target ID

Each chip in the MWCT2xxxS family includes a unique JTAG ID, which must be changed when instantiated with a different die in an SiP. The next table shows the JTAGC ID for this chip.

Table 576. JTAG/Target ID

Chip	PRN	DC	PIN	MIC	IDCODE ID	JTAG ID	Target ID
MWCT2D17S	0	26h (38d)	160h (352d)	Eh (14d)	1	0996_001Dh	0996_001Dh
MWCT2015S	0	26h (38d)	16Ch (364d)	Eh (14d)	1	0996_C01Dh	0996_C01Dh
MWCT2016S	0	26h (38d)	168h (360d)	Eh (14d)	1	0996_801Dh	0996_801Dh
MWCT2D16S	0	26h (38d)	164h (356d)	Eh (14d)	1	0996_401Dh	0996_401Dh

75.5.3.2 JDC

JDC allows you to access two 32-bit data registers by using the JTAG interface and by software running on one of the CPUs in the chip. These registers exchange data between an internal CPU and an external debug tool.

75.5.4 Peripheral IPG debug implementation

Any core can control a peripheral instance individually. You define the core's control over a peripheral during application development.

For peripheral halt, all individual cores have the same capability and are gated by a dedicated MDM_AP core halt register field. For more information, see the configurations defined in MDM_AP register descriptions.

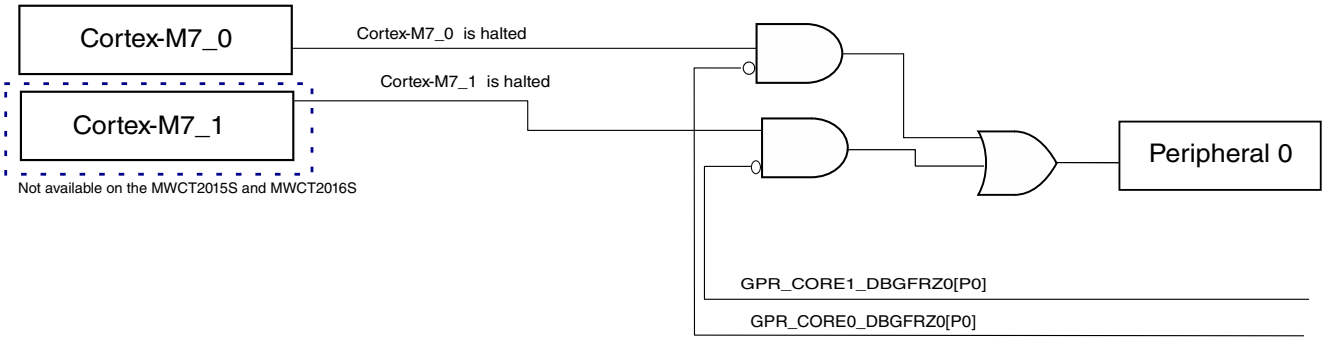


Figure 441. Peripheral IPG debug implementation

75.5.5 Application debugging

This section covers the following two cases:

- Case 1: Debugger connected—application debugging from the first instruction
- Case 2: Debugger not connected

75.5.5.1 Application debugging from first instruction

This chip supports debugging from the first instruction on system power-up, destructive reset, functional reset, and standby exit. However, by default, debugging from the first instruction is disabled.

The next figure shows the timing diagram of application debug implementation from first instruction on system power-up or destructive reset.

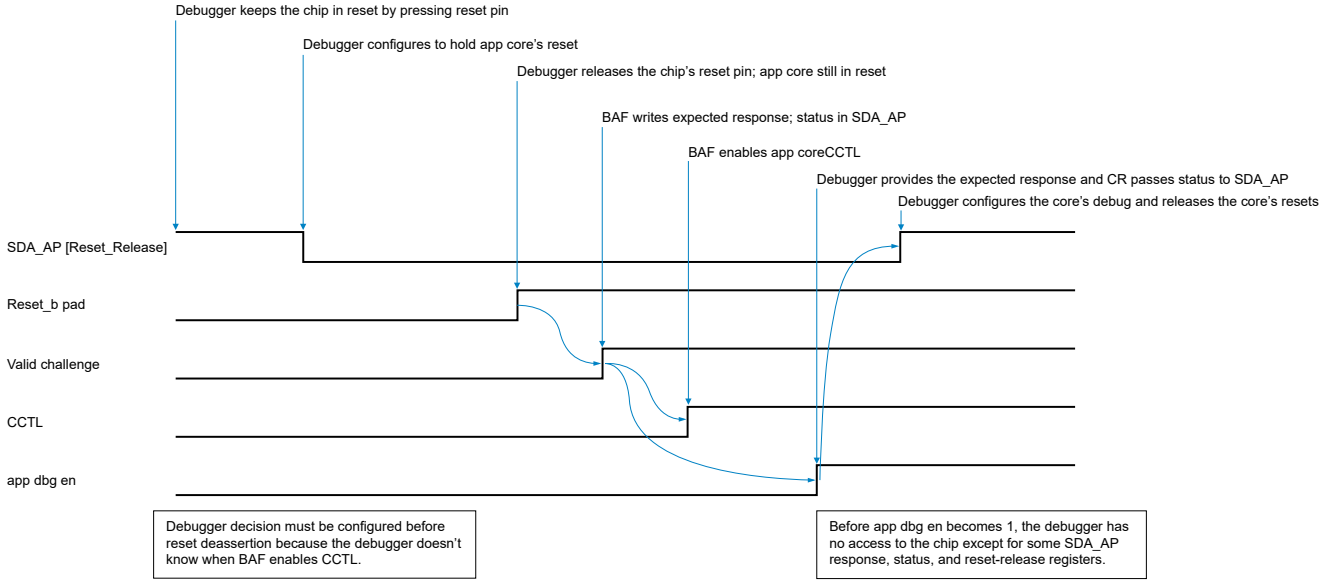


Figure 442. Application debug implementation from first instruction

These are the steps involved in enabling application debug from the first instruction on system power-up or destructive reset:

1. After the reset is applied, the debugger provides an option to debug from the first instruction. To do this:
   a. Write 1 to SDAAPRSTCTRL[RSTRELTLCM70].
   b. Set the reset value of Reset Control (SDAAPRSTCTRL).
2. Cortex M0+ starts after the reset pin is released.
3. Debugger challenge-response (CR) starts.
4. Cortex-M7_0 and Cortex-M7_1 remain in reset until MC_ME's PRTN0_CORE0_PCONF[CCE] field becomes 1 for Cortex-M7_0 and PRTN0_CORE1_PCONF[CCE] field becomes 1 for Cortex-M7_1.
5. When BAF or FW writes 1 to MC_ME's PRTN0_CORE0_PCONF[CCE] field for Cortex-M7_0 and to PRTN0_CORE1_PCONF[CCE] field for Cortex-M7_1, and the debugger CR has passed:
   - If the value of SDAAPRSTCTRL[RSTRELTLCM70] is 0 for the core, the debugger configures the core debug registers and then enables these fields that start the code execution.
   - If the value of SDAAPRSTCTRL[RSTRELTLCM70] is 1 for the core, the debugger configures the core debug registers and then enables these fields that start the code execution.

To debug from the first instruction during the low-power debug protocol, the debugger:

1. Writes 0 to SDAAPRSTCTRL[RSTRELTLCM70] for the cores.
2. Writes 1 to MDMAPWIRREL[WTRSTRGM].

75.5.5.2 Debugger not connected

If the debugger is not connected, the value of the DBGPWRUP_ACK signal is 0 as there is no DBGPWRUP_REQ. This results in the following scenario:

1. The booting core writes 1 to MC_ME's PRTN0_CORE0_PCONF[CCE] field for Cortex-M7_0 and to PRTN0_CORE1_PCONF[CCE] field for Cortex-M7_1.
2. The application core starts running. In case of Lock-step mode, the checker core executes the same code after a delay of two cycles.

75.5.6 Debugger considerations while flash program/erase

The flash programming can be done by application cores as well as debugger. The debugger can program/erase flash using either of the following two options:

1. Loading the program/erase code to SRAM from the debugger and then executing the program/erase sequence by application code from SRAM.
   - Debugger is connected and authenticated.
   - The debugger loads the code in the form of application binary image into the on-chip SRAM.
   - The debugger then initiates the application core to execute the binary from SRAM.
2. Debugger executing the program/erase sequence.
   - Debugger is connected and authenticated.
   - The debugger then initiates the flash programming by reading/writing the registers involved and following the program/erase sequence.

First option is the recommended because of less execution time. In second option, the debugger is the master and since the debug interface is serial, the execution takes more time.

In some cases, you might experience flash program/erase failure due to flash watchdog timeout when debugger executes the flash program/erase sequence due to the serial interface. Therefore, it is recommended to use the first option. In case second option is required, it should be performed in reduced clocking options (Option E and E2 only. For details, refer to the **Clocking details** section in the **Clocking** chapter).

75.5.7 SWJ-DP sequence for debug authentication

To perform the SWJ-DP sequence for debug authentication, the debugger:

1. Polls the AUTHSTTS[CHALRDY] field until it indicates the challenge status.
2. Reads Key Challenge (KEYCHAL0 - KEYCHAL7) if the challenge is valid and creates an authenticated 256-bit response key.
3. Writes a 256-bit response key to Key Response (KEYRESP0 - KEYRESP7).
4. Indicates that the response is ready by configuring AUTHCTL[HSEAUTHREQ].
5. Checks the status of Authentication Status (AUTHSTTS) to evaluate if the operation is successful.

   If the authentication process is successful and the debugger has write access to Debug Enable Control (DBGENCTRL), the challenge or response is considered successful too.

   NOTE

   SDA_AP supports challenge and response based on both JTAG and SWJ-DP modes. See SDAAPGENCTRL0[JTAG_CR_EN] for details.

75.6 APB memory map

You can access the debug registers via the APB_AP bus. The next table shows all the addresses for the CoreSight APB components and the addressing used for accessing the DAP components via the memory interface. You can also access all APB registers from the processing cores.

Table 577. APB components mapping

Debug APB component	APBIC base address (access from debugger)	System memory map address (access from cores)		Memory allocation (KB)	APB-AP0 slot number	Applicability
		Page number	Base address			
APB_AP ROM table	F8h	0	4024_00F8h	4	APBIC base	All
Funnel 0	1000h	0	4024_1000h	4	0	ETM/ ITM:MWCT2D17S ITM: MWCT2016S, MWCT2014S, MWCT2D16S
Funnel 1	2000h	0	4024_2000h	4	1	MWCT2D17S
Funnel 2	3000h	0	4024_3000h	4	2	MWCT2D17S
CM7_cluster_ETF_ETMI	4000h	1	4024_4000h	4	3	MWCT2D17S
CM7_cluster_ETF_ETMD	5000h	1	4024_5000h	4	4	MWCT2D17S
HTM ETF	6000h	1	4024_6000h	4	5	MWCT2D17S
Shared_system_ETF	7000h	1	4024_7000h	4	6	MWCT2D17S
HTM 0	8000h	2	4024_8000h	4	7	MWCT2D17S
HTM 0 CTI	9000h	2	4024_9000h	4	8	MWCT2D17S
TPIU	A000h	2	4024_A000h	4	9	MWCT2D17S
System SWO	B000h	2	4024_B000h	4	10	All
Timestamp CTL	C000h	3	4024_C000h	4	11	All

Funnel 0 should be configure to buffer the data coming from the different sources, to avoid padding data with null packets that affect the bandwidth.

75.7 Trace

75.7.1 Trace modules and connectivity

The Trace subsystem:

- Combines trace data from all internal clients that generate trace information
- Includes a 32-bit TPIU
- Includes these components:
  - ATB
  - ATBR

— CSTF

— Debug APB

— ETF

— TPIU

Multiple options for trace output allow parallel tracing. Trace information can be read from the trace interface or the DAP interface, and traces can be alternatively read out from ETF at a slow speed via APB_AP. Table 578 shows EFT sizes.

Table 578. ETF sizes

FIFO	Memory interface data width (in bits)
Cortex-M7 ETM/ITM cluster ETF	64
Cortex-M7 ETMD cluster ETF	128
HTM ETF	64
Shared system ETF	64

NOTE

- The DMA-HTM trace supports four words at 80 MHz. For HTM trace, both DBGENCTRL[GSPNIDEN] and DBGENCTRL[GSPIDEN] must be 1.
- Enabling any one of data trace causes overflow inside ETM and trace packets get dropped. Enabling instruction trace of both the core simultaneously does not cause any overflow.

Control (MDMAPCTL) provides fields to override the speed control (see Table 579 for details) from some of the trace sinks (and TPIU). The complete trace pipeline bandwidth is limited by the slowest sink component. The default settings of these fields allow maximum bandwidth for the TPIU to trace. When tracing to memory (if supported), the fields may need to be changed.

Table 579. Trace output overrides

Trace destination	SWO_override	TPIU_override
TPIU	1	0

The trace sources of the chip are the cores and their related modules. Trace funnels exist for all possible trace clients. For more information on the various components in the trace bus connectivity, see the *CoreSight Components Technical Reference Manual* (available in References). See Table 580 for details on funnel assignments.

The ATBR is integrated to send a shared funnel output across TPIU. This figure illustrates the chip's detailed trace architecture.

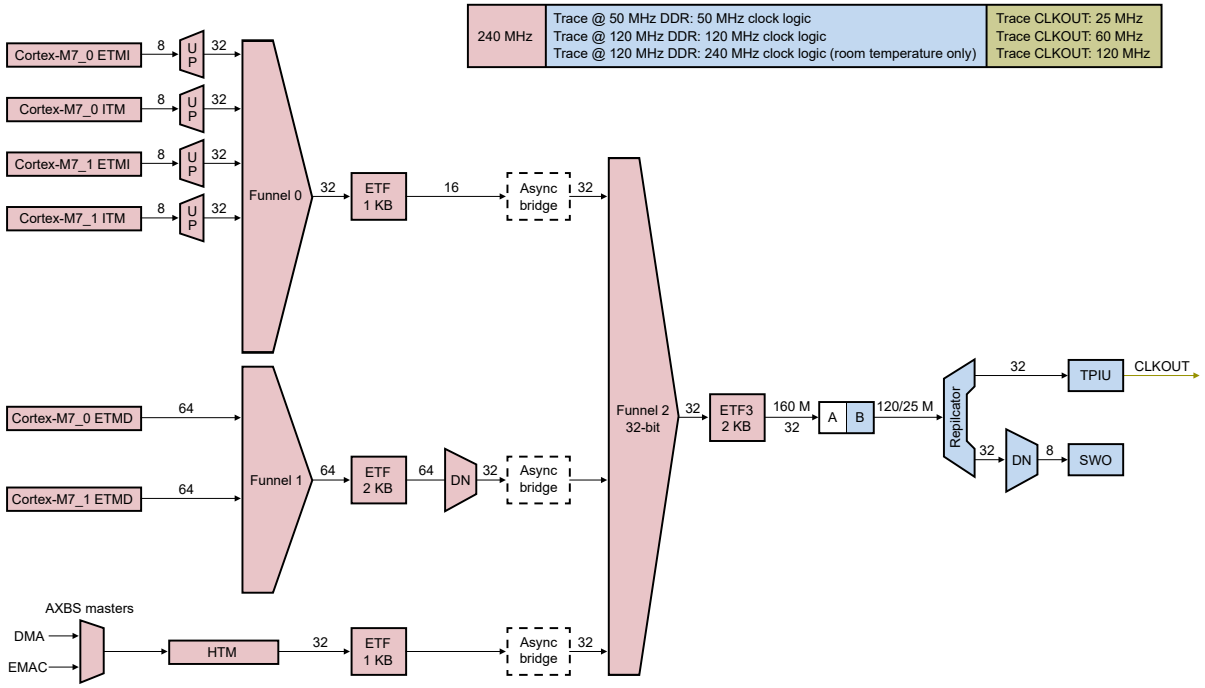


Figure 443. MWCT2D17S detailed trace architecture

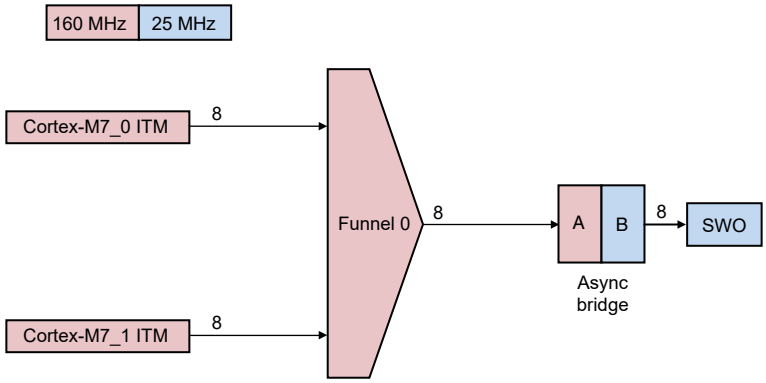


Figure 444. MWCT2D16S detailed trace architecture

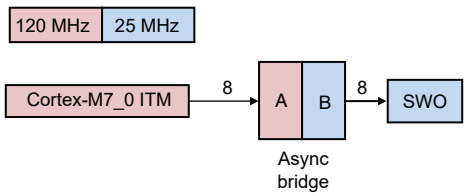


Figure 445. MWCT2016S/MWCT2015S detailed trace architecture

Table 580. Funnel assignments

Funnel	Port	Input	Frequency (MHz)	Applicability
			MWCT2015S/ MWCT2016S	
0	0	Cortex-M7_0 ETMI		MWCT2D17S
0	2	Cortex-M7_1 ETMI		MWCT2D17S
0	3	Cortex-M7_1 ITM		MWCT2D17S
1	0	Cortex-M7_0 ETMD		MWCT2D17S
1	1	Cortex-M7_1 ETMD		MWCT2D17S
1	4-7	Reserved		-
2	0	Cortex-M7 instruction ETF through an asynchronous bridge		MWCT2D17S
2	1	Cortex-M7 data ETF through an asynchronous bridge		MWCT2D17S
2	2	HTM ETF through an asynchronous bridge		MWCT2D17S
2	3-7	Reserved		-

75.7.1.1 Chip's bus trace client

Chips in the MWCT2xxxS family include a bus trace client.

HTM provides the address and data trace information about AXBS buses. The information from an HTM can be used with the debugger to enable easy, accurate debugging on AXBS-based embedded systems. The chip implements an HTM64 configuration to trace 64-bit AXBS masters in the system. To simplify the implementation, instead of tracing the individual ports of various AXBS masters and slaves, which may be running at different frequencies, HTM64 snoops the AXBS crossbar master ports that are all synchronous with a 160 MHz system clock. This table provides details related to the HTM input connectivity.

Table 581. HTM connections

HTM64 port	AHB crossbar port
HTMBUSSELECT0	M0 AXBS_Lite XBIC (DMA)
HTMBUSSELECT1	M3 AXBS (Ethernet)
HTMBUSSELECT2	M6 AXBS (Ethernet 2)

75.7.1.2 TPIU interface

A standard 16-bit parallel TPIU is integrated into the debug subsystem. Chips in the MWCT2xxxS family generate the trace via the trace port.

To prevent instruction trace from being dropped in a multi-core environment, a TPIU throughput of 55.2 MB/s or higher must be maintained. You could use these pin and frequency combinations:

- Four high-speed data + one clock pads @120 MHz (240 Mbit/s throughput per pad) totaling 120 MB/s
- 16 low-speed data + one clock pads @25 MHz (50 Mbit/s throughput per pad) totaling 100 MB/s

High-end chips from the MWCT2xxxS family support at least 56 MB/s of trace throughput on the TPIU interface.

Table 582. Throughput

Core	Throughput value	Applicability
Cortex-M7_0	27.6 MB/s	MWCT2D17S
Cortex-M7_1	27.6 MB/s	MWCT2D17S, MWCT2D16S
Total	82.8 MB/s	

75.7.1.3 TPIU flush

To allow the chip to enter Standby mode, software executes an WFI instruction indicating Standby entry. When the chip enters this mode, MC_RGM asserts a trace flush request to the TPIU and waits for a trace flush done signal from the TPIU before requesting MC_PCU to proceed. The TPIU and debug infrastructure clocks can be gated after this. Because the TPIU is sourced from system clock, the clock must not be gated until this point.

75.7.1.4 Trace through reset

All debug and trace components (HTM, ETM, ETF, and so on) are on destructive reset. When tracing through reset, the clock switches from PLL to FIRC. It is recommended to preserve the following pad configurations across reset, which are performed through MDMAPCTL[DBGRSTFASTPAD] and MDMAPCTL[DBGRSTSLOWPAD]:

- Four high-speed data + one clock pads @120 MHz (240 Mbits/s throughput per pad) totaling 120 MB/s
- 16 low-speed data + one clock pads @25 MHz (50 Mbits/s throughput per pad) totaling 100 MB/s

The implementation involves writing 1 to MDMAPCTL[DBGRSTFASTPAD] or MDMAPCTL[DBGRSTSLOWPAD], depending on which set of trace pads you need to enable. These fields also control:

- The trace pad mux.
- The "obe" of the trace pads, which, if configured, is retained over functional reset because the reset is controlled using the above-mentioned corresponding fields.

75.8 Embedded cross trigger (ECT)

ECT allows multi-core run control and trace cross-triggering, such as synchronous stop-start for all cores or trigger trace on a trigger event from another core or module. See the *CoreSight Components Technical Reference Manual* (available in References) for detailed information on ECT.

ECT architecture involves CTMs and CTIs. The CTIs provide a cross-triggering interface between the cores and other debug and trace modules. The channels of these CTIs are interconnected using CTMs, as shown in this figure.

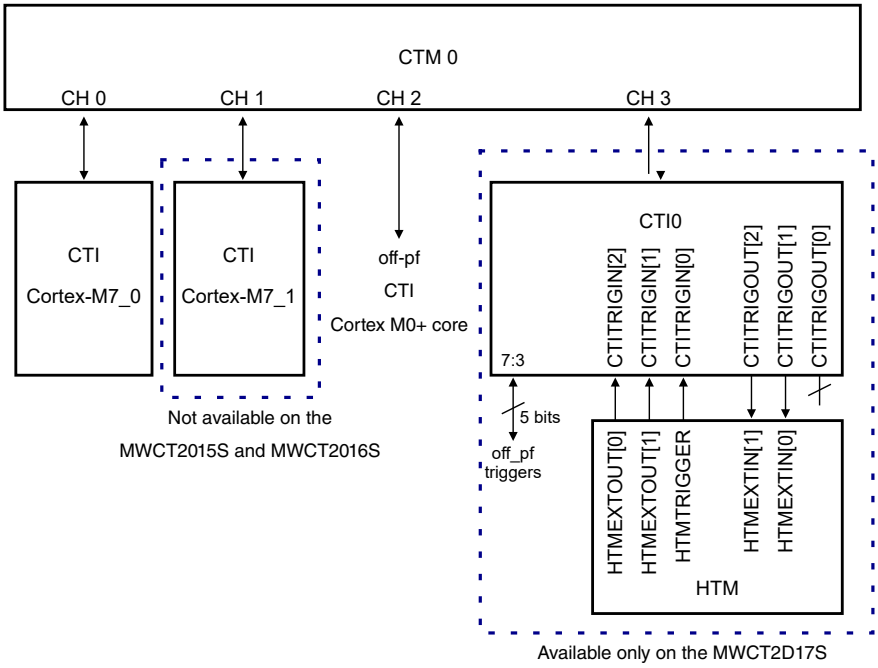


Figure 446. ECT

75.8.1 CTI assignments

Table 583. CTI assignments

CTI instance	Trigger number	Trigger in	Trigger out
CTI_Cortex-M7_0	7	ETM event output 3	Processor restart
	6	ETM event output 2	ETM event input 3
	5	ETM event output 1	ETM event input 2
	4	ETM event output 0	ETM event input 1
	3	DWT comparator output 2	ETM event input 0
	2	DWT comparator output 1	Interrupt request 1
	1	DWT comparator output 0	Interrupt request 0
	0	Processor halted	Processor debug request
CTI_Cortex-M7_1	7	ETM event output 3	Processor restart
	6	ETM event output 2	ETM event input 3
	5	ETM event output 1	ETM event input 2
	4	ETM event output 0	ETM event input 1
	3	DWT comparator output 2	ETM event input 0
	2	DWT comparator output 1	Interrupt request 1

Table continues on the next page...

Table 583. CTI assignments (continued)

CTI instance	Trigger number	Trigger in	Trigger out
	1	DWT comparator output 0	Interrupt request 0
	0	Processor halted	Processor debug request
CTI_Cortex-M7_2	7	ETM event output 3	Processor restart
	6	ETM event output 2	ETM event input 3
	5	ETM event output 1	ETM event input 2
	4	ETM event output 0	ETM event input 1
	3	DWT comparator output 2	ETM event input 0
	2	DWT comparator output 1	Interrupt request 1
	1	DWT comparator output 0	Interrupt request 0
	0	Processor halted	Processor debug request
CTI_Cortex-M7_3	7	ETM event output 3	Processor restart
	6	ETM event output 2	ETM event input 3
	5	ETM event output 1	ETM event input 2
	4	ETM event output 0	ETM event input 1
	3	DWT comparator output 2	ETM event input 0
	1	DWT comparator output 0	Interrupt request 0
	0	Processor halted	Processor debug request
	2	DWT comparator output 1	Interrupt request 1
CTI_0	7	Reserved (grounded)	Reserved (no connection)
	6	ETF_3 full	ETF_3 trigger input
	5	ETF_2 full	ETF_2 trigger input
	4	ETF_1 full	ETF_1 trigger input
	3	ETF_0 full	ETF_0 trigger input
	2	HTM trigger out 2	HTM trigger in 2
	1	HTM trigger out 1	HTM trigger in 1
	0	Reserved	HTM trigger in 0

75.9 Low-power debug handshake protocol

The debugger must perform this sequence to enter or exit Standby mode, if the debugger handshake is enabled:

1. Power on DAP.

   The debugger connection is established.

2. Configure the Debug subsystem for the relevant operations.

3. Write 1 to MDMAPWIREN[LWPWREN] to gate entry into Standby mode with the debugger handshake. If TPIU is disabled, write 1 to MDMAPWIREN[LWPRSTPRVT] too.

4. Disable POR_WDG for monitoring the entry to or exit from Standby mode by writing 1 to DCM's DCMRWP1[8] field. This is required for low-power debug handshake because debugger configurations can be more time consuming than the POR_WDG threshold levels and can raise a false POR_WDG event.
5. Write 1 to DCMRWF1[STANDBY_IO_CONFIG] in the Device Configuration Module (DCM). This causes padkeeping to be disabled on standby entry itself without any software dependency. This is needed in case of low-power debug since otherwise the padkeeping on TDO would result in no debugger communication. In other cases, DCMRWF1[STANDBY_IO_CONFIG] should be configured as 0 before standby entry.
6. Initiate entry into Standby mode. See the "Power Management" chapter for the Standby mode entry sequence and configurations.
7. Identify whether the low-power debug is enabled on the low-power entry.
   - If low-power debug is enabled and TPIU is enabled too, trace flush starts, and the debugger acknowledges the following:
     — Low-power debug traces
     — DAP-related configuration reception and context saving

     After this, the chip enters Standby mode.
   - If TPIU is disabled:
     — The debugger acknowledges DAP-related configuration reception and context saving by writing 1 to MDMAPWIRREL[PRVNTRSTRGM].

     After this, the chip enters Standby mode.
   - If low-power debug is disabled, the chip enters Standby mode without waiting for debugger acknowledgment.

   On any wakeup event, the chip starts the Standby mode exit sequence described in the "Power Management" chapter. After the chip exits Standby mode, the debugger connection is restored. By this time, the debugger is already aware that it has enabled the low-power debug handshake. In this case, the debugger must poll MDMAPSTTS[DESTRST] to check whether:
   - The debug infrastructure is out of reset.
   - The DAP connection can be established.

   After DAP is powered on (the debugger connection is established), the debugger:
   a. Reconfigures the debug and trace attributes using the fields in DBGENCTRL
   b. Writes 1 to MDMAPWIRREL[WTRSTRGM] after the debugger trace context is restored

      The chip exits Standby mode.

      If low-power debug is not configured, the chip exits Standby mode without waiting for the debugger to write to MDMAPWIRREL[WTRSTRGM].
8. Reconfigure the debug and trace configuration in SDA_AP register descriptions after the chip exits Standby mode.

NOTE

If the chip wakes up from Standby mode through pad reset, the debugger must perform a CR again.

75.10 Debug resets

The debug subsystem follows this sequence on the source of reset:

1. POR resets the complete debug logic.
2. The destructive reset resets all types of debug logic except JTAGC.

Conversely, the debug subsystem can generate a system reset using these mechanisms:

- System destructive reset defined in Control (MDMAPCTL) that allows the debugger to provide the destructive reset to the system. The debugger loses connection to the system with this reset.

- System functional reset defined in Control (MDMAPCTL) that allows the debugger to hold the system in functional reset.

To program various debug registers, the functional clocks must be enabled. All debug and trace components must be on destructive reset.

75.11 Debug across device LifeCycles

The debug access to the system is available in early lifecycles and subsequently based on NVM configuration settings.

After the debug is set to 'Trusted', the system access is allowed after a successful authorization step between the debugger and the system.

The authorization provides provisions to allow debug for Cortex M0+ core and Cortex M7 cores.

Following table shows the debug access based on various configuration bits.

Table 584. Debug access based on LifeCycle and bit configurations

LifeCycle	Configuration bits				Debug access
	DBG_EN_OEM	APP_DIS_OEM	DBG_EN_FLD	APP_DIS_FLD	Appl Core Debug
MCU_PROD	-	-	-	-	Open
	-	-	-	-	Open
CUST_DEL	-	-	-	-	Open
	-	-	-	-	Open
OEM_PROD	0	-	-	-	Closed
	1	0	-	-	Trusted
	1	0	-	-	Trusted
	1	1	-	-	Disabled
IN_FIELD	-	-	0	-	Closed
	-	-	1	0	Trusted
	-	-	1	0	Trusted
	-	-	1	1	Disabled
PRE_FA	-	-	-	-	Trusted
	-	-	-	-	Trusted
FA	-	-	-	-	Open

NOTE

1. **Closed:** Debug is not possible in this state (not even if keys are known - no authentication is possible)
2. **Open:** Debug is always possible
3. **Trusted:** Debug is possible after successful authentication (Challenge/Response handshake with correct credentials)
4. **Disabled:** Debug has been explicitly disabled via burning of a fuse in that LC, otherwise behaves as Closed
5. During functional reset, the debug-enable will retain its last status, while DCM scans NVM for lifecycle and DCF values. Debug status will be re-evaluated once dcm_done gets 1.
6. During temporary advancement of lifecycle, debug-en will immediately reflect the status based on updated lifecycle/DCF bits.

75.12 Pin interface

This table presents a summary of functional and power pins that are used for debugging purposes.

Table 585. Pin interface

Pin type	Pins	Number of pins (balls)[1]	Nominal voltage
Functional JTAG	JCOMP, TCK, TMS, TDI, TDO	Five (only TDI and TDO can be multiplexed with GPIO or other functions)	3.3 V, 5 V
Functional trace pins (parallel trace)	TRACE_CLK	One	3.3 V, 5 V
	TRACE_D[15:0]	16	3.3 V, 5 V
Ground	VSS	See the IOMUX file attached to this document for information on the number of VSS pins in various packages.	0 V

1. The terms pins and balls are used interchangeably. Some chip packages include pins and others include balls.

75.12.1 Debug port and pin descriptions

The pads to which the debug signals are mapped operate using the JTAG functionality out of reset but can later be reassigned to their alternate functionalities. TDI and TDO can operate as alternate GPIO functions. See this table for pin assignments in different modes.

Table 586. Debug port pins

Pin name	JTAG debug port		Internal pullup or pulldown logic
	Type	Description	
TMS	I/O	JTAG test mode selection (TMS)	Pullup
TCK/SWCLK	I	JTAG test clock (TCK)	Pulldown
TDI	I	JTAG test data input (TDI)	Pullup
TDO/TRACESWO	O	JTAG test data output (TDO)	Not connected

75.12.2 Trace port pin descriptions

This chip generates trace via TPIU that transmits the trace data out of the chip over a parallel trace port. The trace port consists of an ETM trace clock and 16 parallel trace data outputs. The Arm optional trace port control (TRACECTRL), debug request (DBGRQ), and debug acknowledge (DBGACK) signals are not implemented.

Table 587. Trace output port pins

Pin name	Description
TRACE_DATA00	ETM parallel trace data output 01
TRACE_DATA01	ETM parallel trace data output 01
TRACE_DATA02	ETM parallel trace data output 02
TRACE_DATA03	ETM parallel trace data output 03
TRACE_DATA04	ETM parallel trace data output 04
TRACE_DATA05	ETM parallel trace data output 05
TRACE_DATA06	ETM parallel trace data output 06
TRACE_DATA07	ETM parallel trace data output 07
TRACE_DATA08	ETM parallel trace data output 08
TRACE_DATA09	ETM parallel trace data output 09
TRACE_DATA10	ETM parallel trace data output 10
TRACE_DATA11	ETM parallel trace data output 11
TRACE_DATA12	ETM parallel trace data output 12
TRACE_DATA13	ETM parallel trace data output 13
TRACE_DATA14	ETM parallel trace data output 14
TRACE_DATA15	ETM parallel trace data output 15
TRACE_CLK	ETM parallel trace clock output

NOTE

By default, Rx pins float and are not pulled inside. An internal active pulldown logic exists only when you enable Rx via software (IBE).

75.13 Timestamp distribution network

The timestamp distribution network uses CoreSight timestamp components to generate a 48-bit timestamp value for the trace sources, as shown in the next figure. The CoreSight timestamp generator generates a 64-bit counter value, but only the least significant 48 bits are distributed to the trace sources. A 7-bit narrow timestamp is derived from the 48-bit timestamp and distributed to the trace client, where a decoder regenerates the 48-bit timestamp. The generator must be programmed, and you could find the related programming information in the *CoreSight Components Technical Reference Manual* (available in References).

This chip supports 48-bit timestamping. The generator operates at a frequency of up to 320 MHz and gives a 64-bit timestamp value..

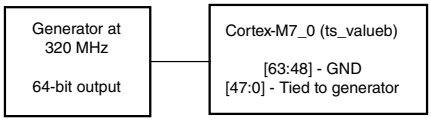


Figure 447. Timestamp distribution network

75.14 Peripheral debug freeze register descriptions

Implement the logic provided in this section for each peripheral instance supporting the debug operation.

NOTE

Debug freeze is enabled for all the peripherals so that as soon as the core halts, peripherals can be frozen to support debugging from the first instruction.

For register details, refer to the following registers in the Device Configuration Module (DCM):

- Read Write GPR On Destructive Reset Register (DCMRWD6)
- Read Write GPR On Destructive Reset Register (DCMRWD7)
- Read Write GPR On Destructive Reset Register (DCMRWD8)
- Read Write GPR On Destructive Reset Register (DCMRWD9)

75.15 MDM_AP register descriptions

The debugger has access to the status and control elements implemented as registers in MDM_AP, which is selected by APSEL (6h) on the DAP bus. These registers provide additional control and status information for typical debug, cross-triggering, and run-control scenarios. Also, the register fields provide a way for the debugger to get the updated status of the core without initiating a bus transaction across the crossbar switch, thus remaining less intrusive during a debug session.

MDM_AP is accessible as DAP (see DAP TAP for details).

75.15.1 MDM_AP memory map

MDM_AP base address: 4025_0600h

Offset	Register	Width (In bits)	Access	Reset value
0h	Status (MDMAPSTTS)	32	R	7000_0000h
4h	Control (MDMAPCTL)	32	RW	0640_0000h
30h	WIR Enable (MDMAPWIREN)	32	RW	0000_0000h
34h	WIR Status (MDMAPWIRSTTS)	32	R	0000_0000h
38h	WIR Release (MDMAPWIRREL)	32	RW	0000_0000h

75.15.2 Status (MDMAPSTTS)

Offset

Register	Offset
MDMAPSTTS	0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	CM71 DBG...	CM70 DBG...	Reserved				Reserved	Reserved	CM71 SLP...	CM70 SLP...	Reserved	Reserved	CM71 DPS...	CM70 DPS...
W																
Reset	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	CM71 HLT	CM70 HLT	Reserved									FUNC RST	DEST RST	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 CM71DBGRSTRD	Cortex-M7_1 Debug Restarted Indicates if Cortex-M7_1 has returned to Normal mode from Debug mode. 0b - In Debug mode 1b - In Normal mode
28 CM70DBGRSTRD	Cortex-M7_0 Debug Restarted Indicates if Cortex-M7_0 has returned to Normal mode from Debug mode. 0b - In Debug mode 1b - In Normal mode
27-24 —	Reserved
23	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
22 —	Reserved
21 CM71SLPNG	CM7_1 Sleeping Indicates if Cortex-M7_1 is in Sleep mode. 0b - Not in Sleep mode 1b - In Sleep mode
20 CM70SLPNG	Cortex-M7_0 Sleeping Indicates if Cortex-M7_0 is in Sleep mode. 0b - Not in Sleep mode 1b - In Sleep mode
19 —	Reserved
18 —	Reserved
17 CM71DPSLP	Cortex-M7_1 Deep Sleep Indicates if Cortex-M7_1 is in Deep Sleep mode. 0b - Not in Deep Sleep mode 1b - In Deep Sleep mode
16 CM70DPSLP	Cortex-M7_0 Deep Sleep Indicates if Cortex-M7_0 is in Deep Sleep mode. 0b - Not in Deep Sleep mode 1b - In Deep Sleep mode
15 —	Reserved
14 —	Reserved
13 CM71HLT	CM7_1 Debug Halted Indicates if Cortex-M7_1 is halted because of entry into Debug mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is reserved and not applicable for MWCT2014S, MWCT2015S, and MWCT2016. Reading this field on these devices returns zero. 0b - Core is not halted 1b - Core is halted
12 CM70HLT	Cortex-M7_0 Halted Indicates if Cortex-M7_0 is halted because of entry into Debug mode. 0b - Core is not halted 1b - Core is halted
11-3 —	Reserved
2 FUNCRST	Functional Reset Indicates the system reset state. 0b - Not in functional reset 1b - In functional reset
1 DESTRST	Destructive Reset Indicates the system reset state. 0b - Not in destructive reset 1b - In destructive reset
0 —	Reserved

75.15.3 Control (MDMAPCTL)

Offset

Register	Offset
MDMAPCTL	4h

Function

Allows the debugger:

- To give the destructive reset to the system. A system destructive reset enables this. The debugger also loses connection to the system with this reset.
- To hold the system in functional reset. A system functional reset enables this.

NOTE

The trace functionality on trace pins is selected with this register and is retained across the functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	CM71DBG...	CM70DBG...	Reserved					SWOOVRD	Reserved	TRIUOVRD	Reserved		CM7_1_C...	CM7_0_C...
W																
Reset	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POR_WDG...	Reserved	DBGRSTF...	DBGRSTS...	Reserved	Reserved	CM71DBG...	CM70DBG...	Reserved		SYSFUNC...	SYSRESE...	Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 CM71DBGRSRT	Cortex-M7_1 Debug Restart Indicates if a request to Cortex-M7_1 to leave the debug halt state is asserted. 0b - Normal operation 1b - Request asserted
28 CM70DBGRSRT	Cortex-M7_0 Debug Restart Indicates if a request to Cortex-M7_0 to leave the debug halt state is asserted. 0b - Normal operation 1b - Request asserted
27-23 —	Reserved
22 SWOOVRD	SWO Override Indicates if the SWO trace response is overriden. When SWO is not the selected trace sink target, you must override the trace response. 0b - Not overridden, and SWO generates the trace response 1b - Is overridden

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved
20 TRIUOVRD	TPIU Override Indicates if TPIU trace response is overridden. When TPIU is not the selected trace sink target, you must override the trace response. 0b - Not overridden, and TPIU generates the trace response 1b - Is overridden and asserted
19-18 —	Reserved
17 CM7_1_CORE_ACCESS	Debugger Access To Application Cortex-M7_1 Indicates if debugger access to Cortex-M7_1 across the functional reset phase is supported. After programming the Cortex-M7_1 core debug logic, the debugger can write 1 to this field. Because of this, the clock gating control for CLK and FCLK of Cortex-M7_1 shifts to CCTL. 0b - Supported 1b - Not supported
16 CM7_0_CORE_ACCESS	Debugger Access To Application Cortex-M7_0 Indicates if debugger access to Cortex-M7_0 across the functional reset phase is supported. After programming the Cortex-M7_0 core debug logic, the debugger can write 1 to this field. Because of this scenario, the clock gating control for CLK and FCLK of Cortex-M7_0 shifts to CCTL. 0b - Supported 1b - Not supported
15 POR_WDG_DIS	Power Watchdog Status When a 0 is written to this field, the power watchdog is disabled in case of MDM FUNC reset or JTAG reset. When a 1 is written to this field, the power watchdog is enabled in case of MDM FUNC reset or JTAG reset. 0b - Power watchdog is disabled 1b - Power watchdog is enabled
14 —	Reserved
13 DBGRSTFASTPAD	Debug Over Reset Via Fast Pads Enables or disables trace via fast pads. If enabled, the trace pads have trace over functional reset feature enabled. This field does not take care of the clock configuration.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Pad configuration must be DSE = 1b and SRE = 0b. 0b - Disabled 1b - Enabled
12 DBGRSTSLOWPAD	Debug Over Reset Via Slow Pads Enables or diasables trace via slow pads. If enabled, the trace pads have trace over functional reset feature enabled. This field does not take care of the clock configuration. 0b - Disabled 1b - Enabled
11 —	Reserved
10 —	Reserved
9 CM71DBGREQ	Cortex-M7_1 Debug Request Drives the EDBGREQ input for Cortex-M7_1 and indicates if the debug request is generated. When the core goes into debug state, the field acknowledges that with a halted output signal (see MDMAPSTTS[CM71HLT]). If the core is in Stop mode, this field is used to wake up the core and transition it to the debug halt state. 0b - Debug request is not generated 1b - Debug request is generated
8 CM70DBGREQ	Cortex-M7_0 Debug Request Drives the EDBGREQ input for Cortex-M7_0 and indicates if the debug request is generated. When the core goes into debug state, the field acknowledges that with a halted output signal (see MDMAPSTTS[CM70HLT]). If the core is in Stop mode, this field is used to wake up the core and transition it to the debug halt state. 0b - Debug request is not generated 1b - Debug request is generated
7-6 —	Reserved
5 SYSFUNCRST	System Functional Reset Asserts or deasserts functional reset to the chip. The debugger maintains connection with the chip. 0b - Deasserted 1b - Asserted

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SYSRESETREQ	System Destructive Reset Asserts or deasserts destructive reset to the chip. The debugger also loses connection with this reset. When this field is reset, the entire system comes out of reset. 0b - Deasserted 1b - Asserted
3-0 —	Reserved

75.15.4 WIR Enable (MDMAPWIREN)

Offset

Register	Offset
MDMAPWIREN	30h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														LWPRSTP...	LWPWREN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	Reserved
1 LWPRSTPRVT	Low-Power Entry 0b - Prevents MC_RGM from generating the reset until the MC_RGM TPIU flush receives an acknowledgment from TPIU 1b - Prevents MC_RGM from generating the reset even if MC_RGM receives an acknowledgment from TPIU

Table continues on the next page...

Table continued from the previous page...

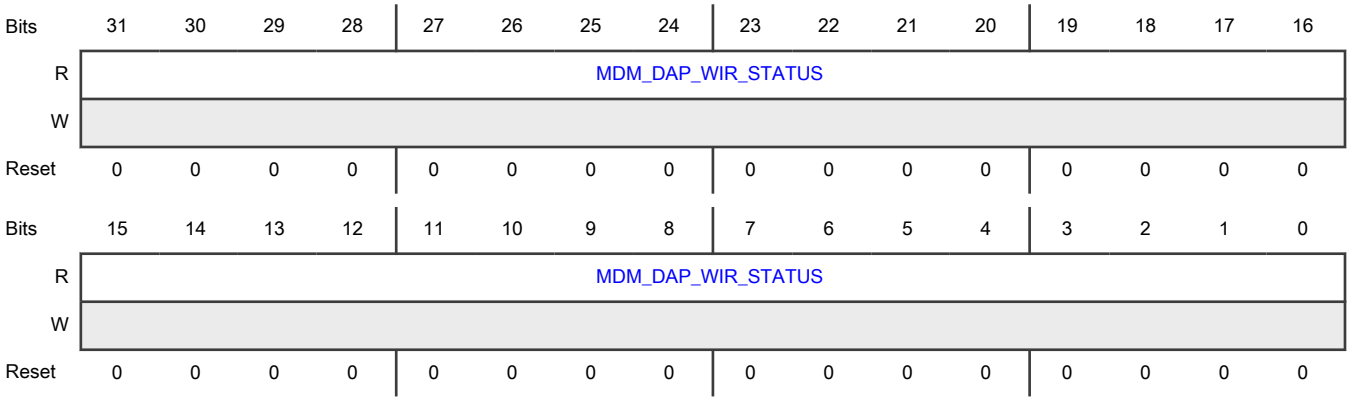
Field	Function
0 LWPWREN	Low Power Debug Enable Enables or disables low-power debug. 0b - Disabled 1b - Enabled

75.15.5 WIR Status (MDMAPWIRSTTS)

Offset

Register	Offset
MDMAPWIRSTTS	34h

Diagram



Fields

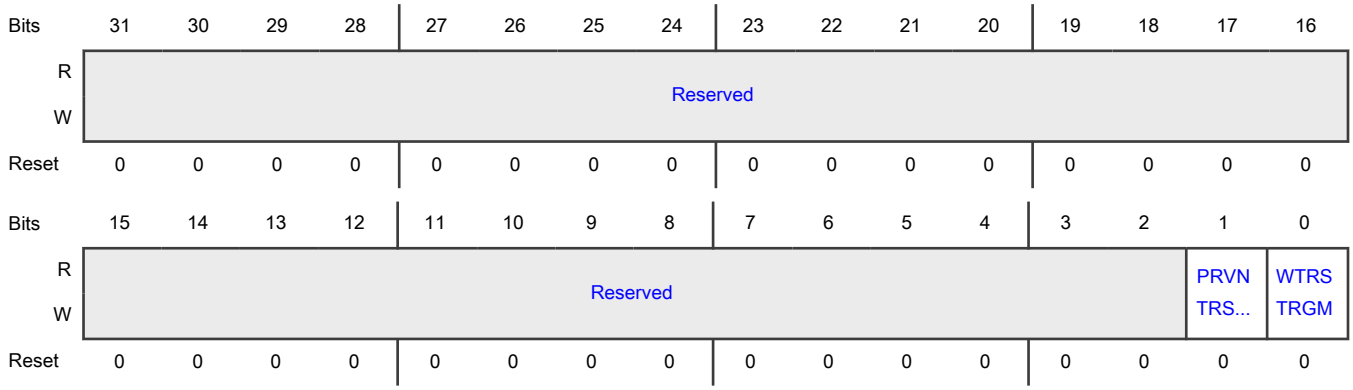
Field	Function
31-0 MDM_DAP_WIR_STATUS	MDM_AP WIR Status 0b - Indicates WIR status 1b - Does not indicate WIR status

75.15.6 WIR Release (MDMAPWIRREL)

Offset

Register	Offset
MDMAPWIRREL	38h

Diagram



Fields

Field	Function
31-2 —	Reserved
1 PRVNTRSTRGM	Prevent Reset Indicates if MC_RGM is prevented from generating reset. After TPIU flush, this field prevents MC_RGM from generating reset even if MC_RGM receives an acknowledge response from TPIU. This is valid for low-power entry. 0b - Normal operation 1b - MC_RGM prevented
0 WTRSTRGM	Wait In Reset B Indicates if waiting of MC_RGM from generating reset is supported. On exiting Standby mode, MC_RGM waits until the debugger writes to another field in the MDM_AP register to allow it to exit reset. 0b - Normal operation 1b - Wait supported

75.16 SDA_AP register descriptions

The debugger and system have access to secure authentication control and status, implemented as registers in SDA_AP on the DAP bus. These registers provide authentication control, authentication status, key exchange information, and debug enable controls.

75.16.1 SDA_AP memory map

SDA_AP base address: 4025_4700h

Offset	Register	Width (In bits)	Access	Reset value
0h	Authentication Status (AUTHSTTS)	32	R	6000_0004h
4h	Authentication Control (AUTHCTL)	32	RW	0000_0000h
10h - 2Ch	Key Challenge (KEYCHAL0 - KEYCHAL7)	32	R	0000_0000h
40h - 5Ch	Key Response (KEYRESP0 - KEYRESP7)	32	RW	0000_0000h
70h	User Identification 0 (UID0)	32	R	0000_0000h
74h	User Identification 1 (UID1)	32	R	0000_0000h
80h	Debug Enable Control (DBGENCTRL)	32	RW	See section
90h	Reset Control (SDAAPRSTCTRL)	32	RW	0600_0000h
A0h	SDA_AP Generic Status (SDAAPGENSTATUS0)	32	R	0000_0000h
A4h	Generic Control 0 (SDAAPGENCTRL0)	32	RW	0000_0000h
B0h	SDA_AP Generic Status (SDAAPGENSTATUS1)	32	R	0000_0000h
C0h	SDA_AP Generic Status (SDAAPGENSTATUS2)	32	R	0000_0000h
D0h	SDA_AP Generic Status (SDAAPGENSTATUS3)	32	R	0000_0000h
E0h	SDA_AP Generic Status (SDAAPGENSTATUS4)	32	R	0000_0000h
FCh	Identity (ID)	32	R	001C_0040h

75.16.2 Authentication Status (AUTHSTTS)

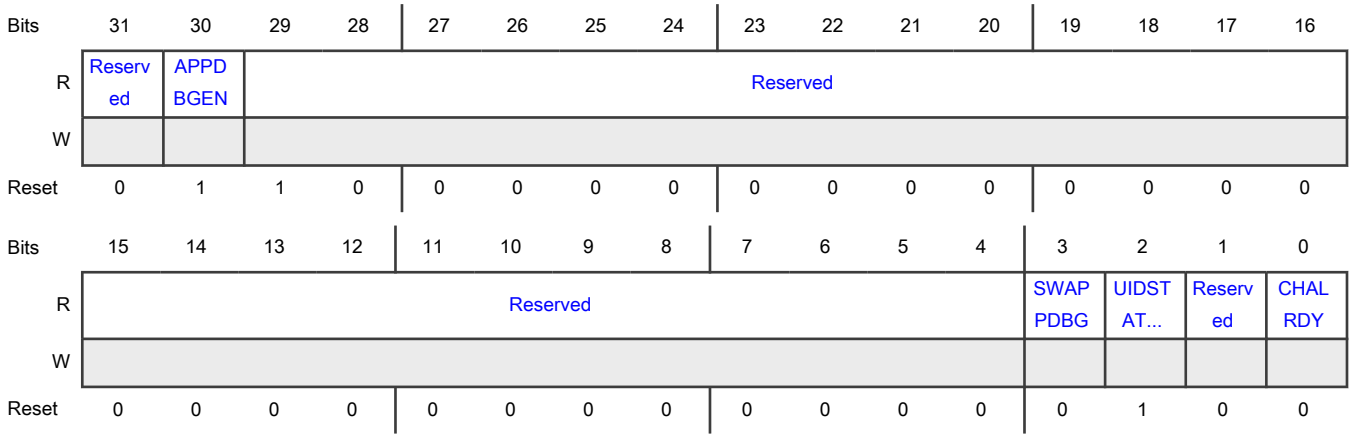
Offset

Register	Offset
AUTHSTTS	0h

Function

Indicates the status of the authentication process.

Diagram



Fields

Field	Function
31 —	Reserved
30 APPDBGEN	Application Debug Enabled or Disabled Indicates: • The status of application debug • Whether CR is satisfied • Whether access to other APs is allowed 0b - Application debug disabled 1b - Application debug enabled
29-4 —	Reserved
3 SWAPPDBG	Software Application Debug Indicates: • The status of software application debug • Whether access to debug controls is allowed 0b - Software application debug disabled 1b - Software application debug enabled
2 UIDSTATUS	User Identification Status Indicates: • The status of UID • Whether DCM has finished reading the flash memory user section

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - UID is not ready and is invalid 1b - UID is ready and is valid
1 —	Reserved
0 CHALRDY	Challenge Ready Indicates: • The status of challenge ready when the value of export control is 0 • The status of the DCM_DONE signal if the value of export control is 1 0b - Challenge is not ready 1b - Challenge is ready

75.16.3 Authentication Control (AUTHCTL)

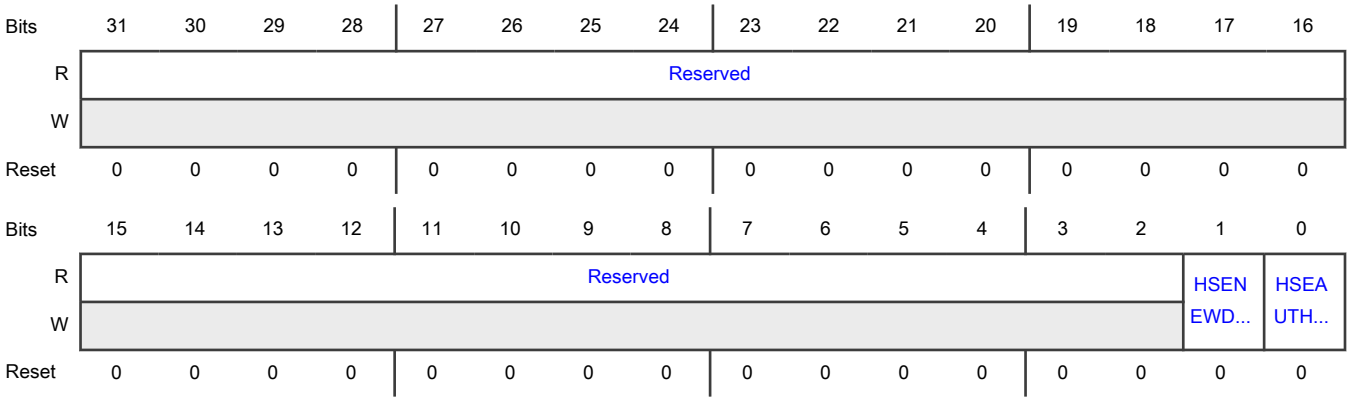
Offset

Register	Offset
AUTHCTL	4h

Function

Controls the authentication process.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 HSENEWDATACTL	New Data Control Indicates that the debugger has consumed the data registers. It is alright for the core to provide new data. 0b - Does not indicate that the debugger has consumed the data registers 1b - Indicates that the debugger has consumed the data registers
0 HSEAUTHREQ	Debug Enablement Authentication Request Indicates that all key values are written and the chip can start the authentication request. 0b - Does not start the authentication request 1b - Starts the authentication request

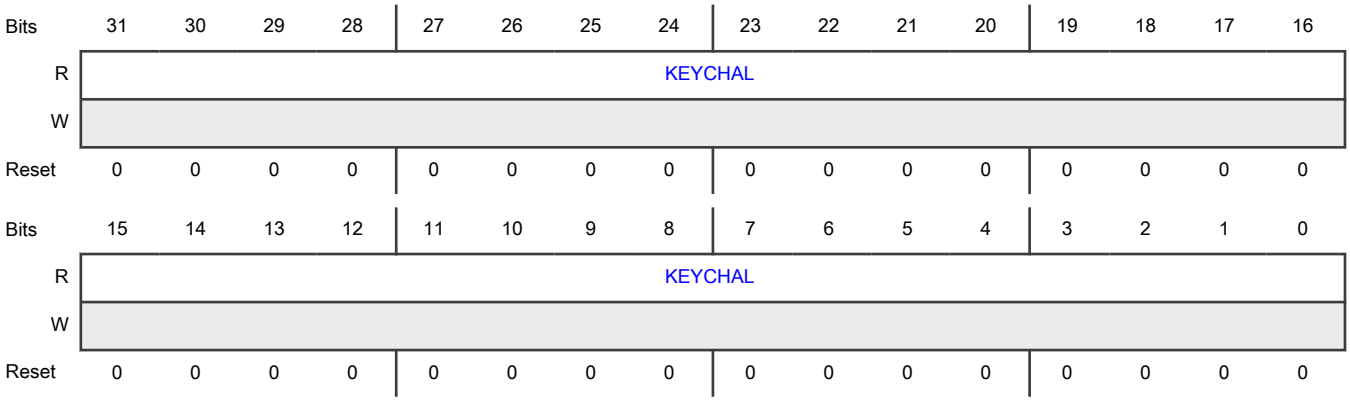
75.16.4 Key Challenge (KEYCHAL0 - KEYCHAL7)

Offset

For a = 0 to 7:

Register	Offset
KEYCHALa	10h + (a × 4h)

Diagram



Fields

Field	Function
31-0	Debug Enablement Key Challenge

Table continues on the next page...

Field	Function
KEYCHAL	

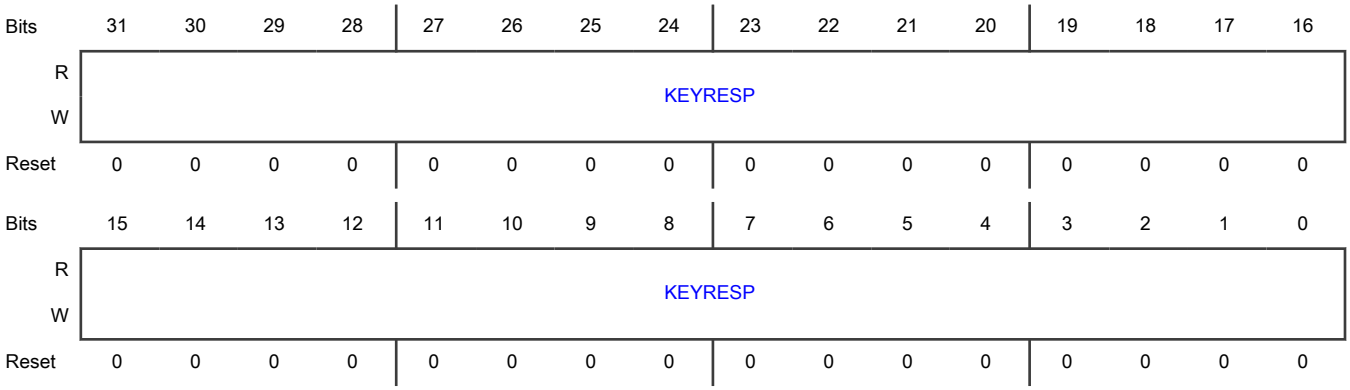
75.16.5 Key Response (KEYRESP0 - KEYRESP7)

Offset

For a = 0 to 7:

Register	Offset
KEYRESPa	40h + (a × 4h)

Diagram



Fields

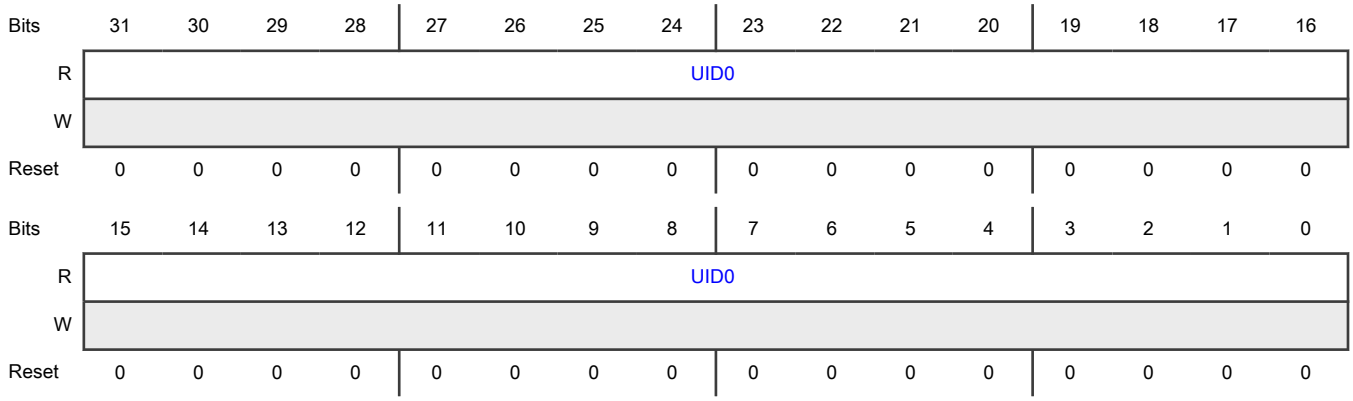
Field	Function
31-0 KEYRESP	Debug Enablement Key Response

75.16.6 User Identification 0 (UID0)

Offset

Register	Offset
UID0	70h

Diagram



Fields

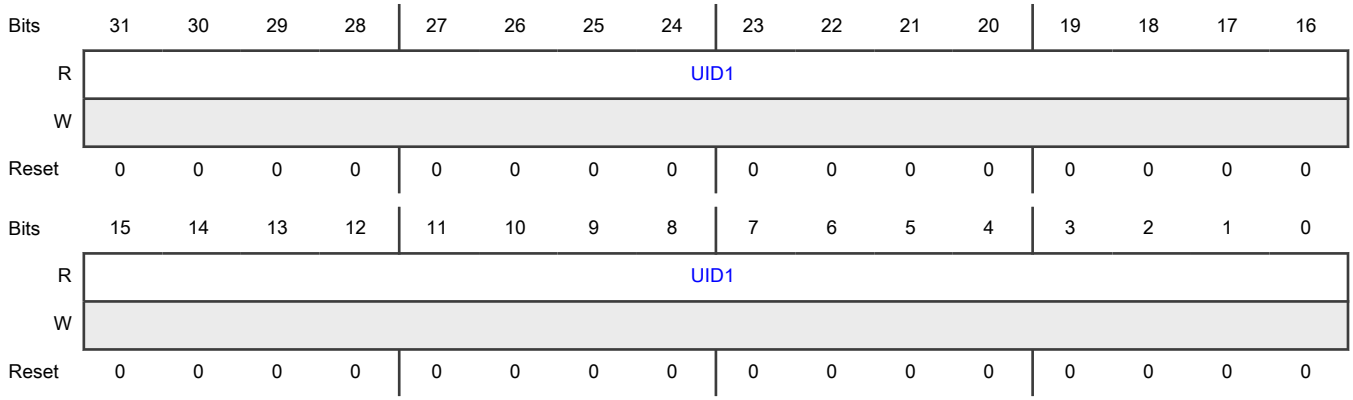
Field	Function
31-0 UID0	User ID 0 Indicates the JTAG user ID bits of the lower word.

75.16.7 User Identification 1 (UID1)

Offset

Register	Offset
UID1	74h

Diagram



Fields

Field	Function
31-0 UID1	User ID 1 Indicates the JTAG user ID bits of the upper word.

75.16.8 Debug Enable Control (DBGENCTRL)

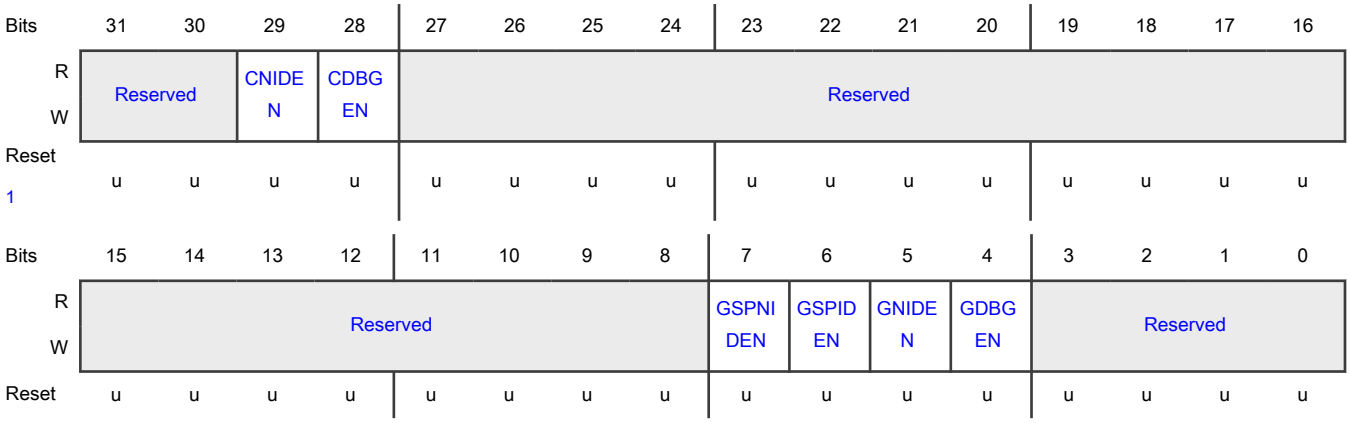
Offset

Register	Offset
DBGENCTRL	80h

Function

Includes a special protection that allows access from Cortex-M0+ only if bit 30 of Authentication Status (AUTHSTTS) is 1.

Diagram



1. The reset value is controlled by an export control enable input. If export control is enabled, the reset value of this register is FFFF_FFF0h. Otherwise, it is 0000_0000h.

Fields

Field	Function
31-30 —	Reserved
29 CNIDEN	Core Non-Invasive Debug Enable Controls CNIDEN of debug blocks coupled with the Cortex-M7 core. 0b - Disabled 1b - Enabled
28 CDBGEN	Core Debug Enable Controls CDBGEN of debug blocks coupled with the Cortex-M7 core. 0b - Disabled 1b - Enabled
27-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

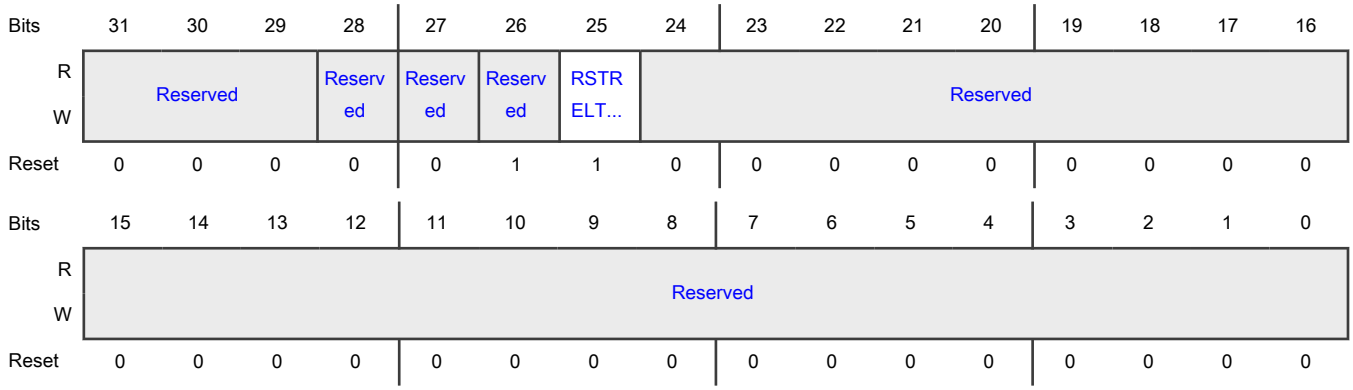
Field	Function
7 GSPNIDEN	Global Secure Privileged Non-Invasive Debug Enable Controls GSPNIDEN of debug blocks coupled with Cortex-M7's subsystems, ETM, ITM, and CTI. 0b - Disabled 1b - Enabled
6 GSPIDEN	Global Secure Privileged Debug Enable Controls GSPIDEN of debug blocks coupled with Cortex-M7's subsystems, ETM, ITM, and CTI. 0b - Disabled 1b - Enabled
5 GNIDEN	Global Non-Invasive Debug Enable Controls GNIDEN of debug blocks coupled with Cortex-M7's subsystems, ETM, ITM, and CTI. 0b - Disabled 1b - Enabled
4 GDBGEN	Global Debug Enable Controls GDBGEN of debug blocks coupled with Cortex-M7's subsystems, ETM, ITM, and CTI. 0b - Disabled 1b - Enabled
3-0 —	Reserved

75.16.9 Reset Control (SDAAPRSTCTRL)

Offset

Register	Offset
SDAAPRSTCTRL	90h

Diagram



Fields

Field	Function
31-29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 RSTRELTLCM70	Reset Release Cortex-M7_0 Indicates if the control signal released the reset for Cortex-M7_0. The reset is released to debug the core from the first instruction. The default value of this field is 1. 0b - Core is in reset 1b - Reset is released
24-0 —	Reserved

75.16.10 SDA_AP Generic Status (SDAAPGENSTATUS0 - SDAAPGENSTATUS4)

Offset

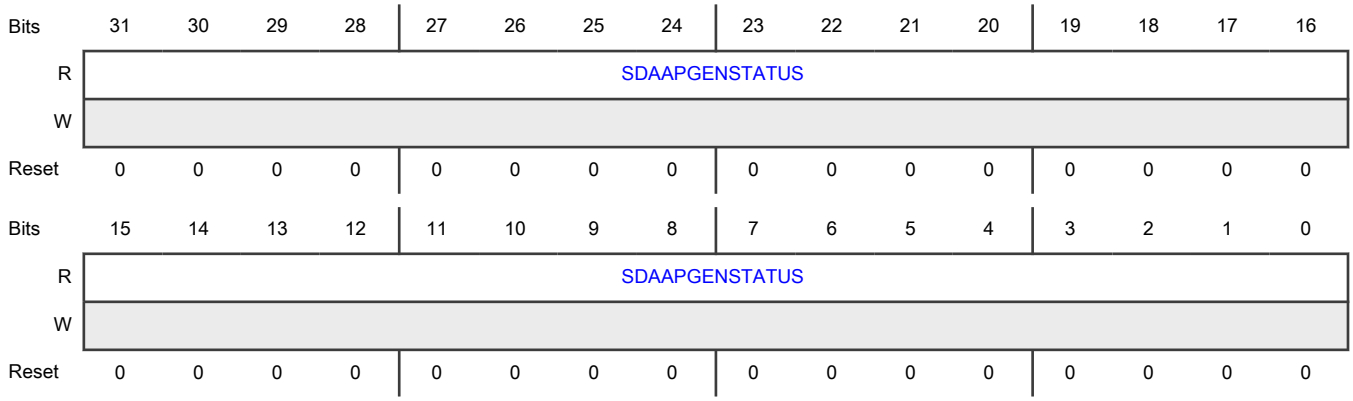
Register	Offset
SDAAPGENSTATUS0	A0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
SDAAPGENSTATUS1	B0h
SDAAPGENSTATUS2	C0h
SDAAPGENSTATUS3	D0h
SDAAPGENSTATUS4	E0h

Diagram



Fields

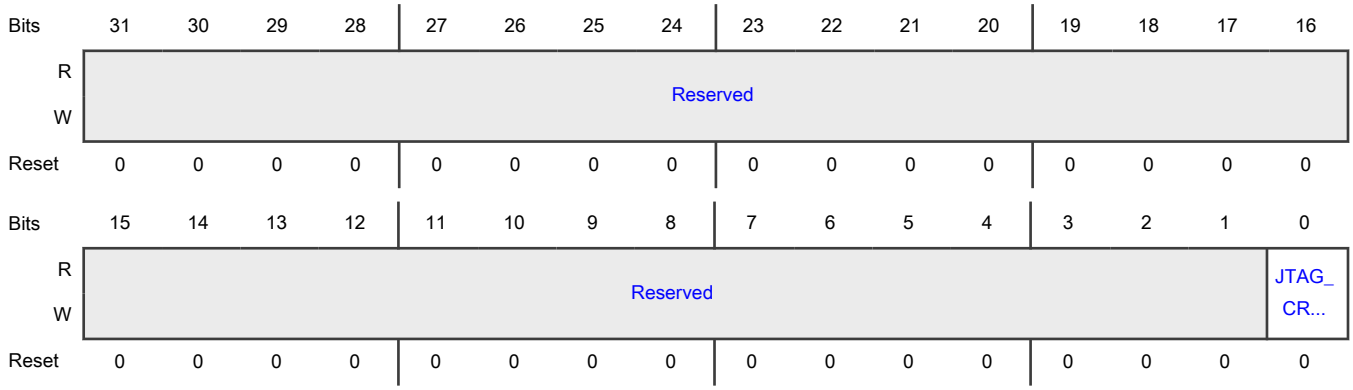
Field	Function
31-0 SDAAPGENSTATUS	DAP Generic Status Is a generic status field for future use. 0b - Does not show generic status 1b - Shows generic status

75.16.11 Generic Control 0 (SDAAPGENCTRL0)

Offset

Register	Offset
SDAAPGENCTRL0	A4h

Diagram



Fields

Field	Function
31-1 —	Reserved
0 JTAG_CR_EN	JTAG CR Enable Performs CR or password comparison based on SWJ-DP mode or JTAG mode. If you write 1 to this field, this function is performed using JTAG irrespective of SWJ-DP mode of the debugger. 0b - Function performed on the basis of SWJ-DP mode 1b - Function performed on the basis of JTAG mode

75.16.12 Identity (ID)

Offset

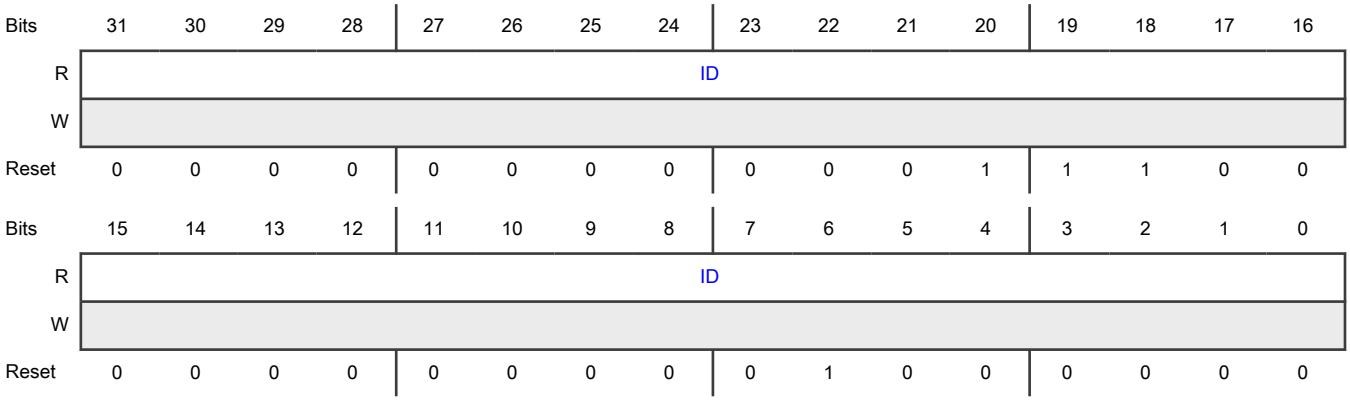
Register	Offset
ID	FCh

Function

NOTE

This register does not generate the bus error on performing the write operation.

Diagram



Fields

Field	Function
31-0 ID	Identity

75.17 Glossary

AHB	Advanced high-performance bus
AHB_AP	Advanced high-performance bus access port
APB	Advanced peripheral bus
APB_AP	Advanced peripheral bus access port
ATB	Advanced trace bus
ATBR	ATB replicator
CSTF	CoreSight trace funnel
DAP	Debug access port
DC	Design center
ETF	Embedded CoreSight funnels
JTAG-DP	JTAG debug port
MDM_AP	Miscellaneous debug module access port
MIC	Manufacturer identity code
PIN	Part identification number
PRN	Part revision number
SDA_AP	Serial data access port
SiP	System-in-package
SWJ-DP	Serial wire/JTAG debug port
TAP	Test and debug access port
TPIU	Trace port interface unit

75.18 References

- Arm CoreSight SoC-400 Technical Reference Manual

  https://static.docs.arm.com/100536/0302/coresight_soc400_technical_reference_manual_100536_0302_01_en.pdf
- Arm CoreSight Architecture Specification

  http://infocenter.arm.com/help/topic/com.arm.doc.ihi0029d/IHI0029D_coresight_architecture_spec_v2_0.pdf
- CoreSight Components Technical Reference Manual

  http://infocenter.arm.com/help/topic/com.arm.doc.ddi0314h/DDI0314H_coresight_components_trm.pdf
- Arm Debug Interface Architecture Specification

  https://static.docs.arm.com/ihi0031/c/IHI0031C_debug_interface_as.pdf

Chapter 76
JTAG Controller (JTAGC)

76.1 Chip-specific JTAGC information

The following figure shows the detailed TAP connectivity. SWD/JTAG SELECT, SWDP and JTAG-DP are part of ARM DAP

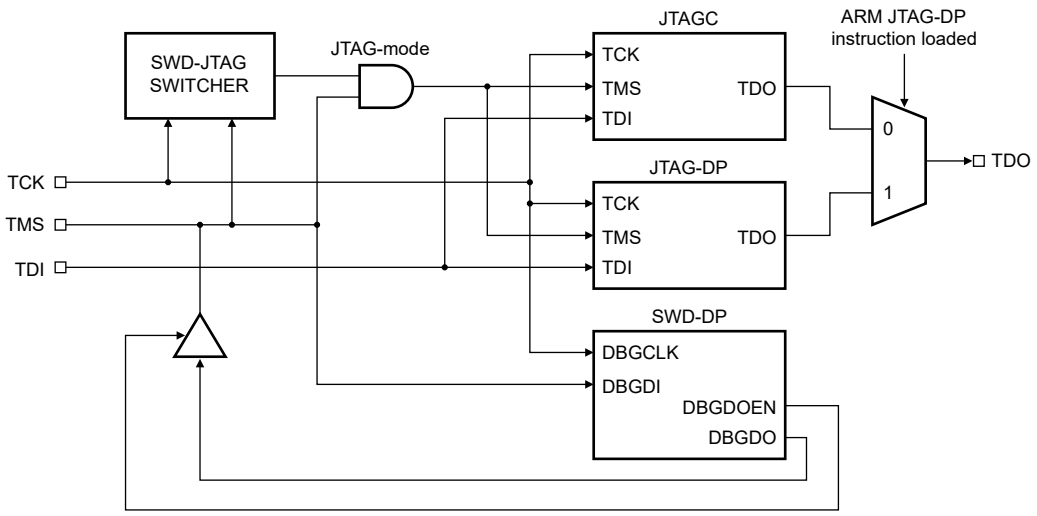


Figure 448. Test and Debug Access Port (TAP) connectivity

The debug port comes out of reset in a standard JTAG mode. It is switched to SWD mode by the change sequences described in JTAG to SWD change sequence . Once the mode has been changed, unused debug pins can be reassigned to any of their alternative muxed functions

JTAG-to-SWD change sequence:

1. Send more than 50 TCK cycles with TMS (SWDIO) =1
2. Send the 16-bit sequence on TMS (SWDIO) = 0111_1001_1110_0111 (MSB transmitted first)
3. Send more than 50 TCK cycles with TMS (SWDIO) =1

NOTE

In case of any reset event, JTAGC is moved to Test Logic Reset (TLR) state first and then JTAGC should be used.

NOTE

If JTAG pins are used for alternative functionality and reset event arrives, then these pins need to be reconfigured to required functionality before using it, in order to avoid unintentional entry in JTAG mode.

76.2 Overview

The JTAGC block provides the means to test chip functionality and connectivity while remaining transparent to system logic when not in test mode. Testing is performed via a boundary scan technique, as defined in the IEEE 1149.1-2001 standard. All data input to and output from the JTAGC block is communicated in serial format.

76.2.1 Block diagram

The following is a simplified block diagram of the JTAG Controller (JTAGC) block. See the chip-specific configuration information within this chapter as well as Register description for more information about the JTAGC registers.

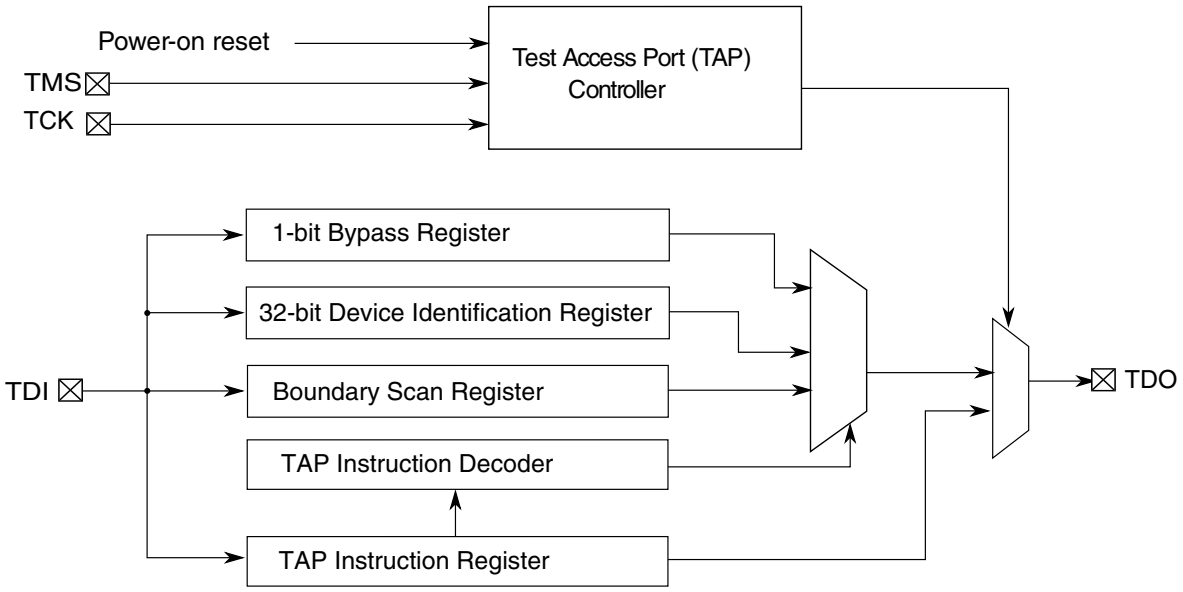


Figure 449. JTAG (IEEE 1149.1) block diagram

76.2.2 Features

The JTAGC block is compliant with the IEEE 1149.1-2001 standard, and supports the following features:

- IEEE 1149.1-2001 TAP interface
  - Four pins (TDI, TMS, TCK, and TDO)
- Instruction register that supports several IEEE 1149.1–2001 defined instructions as well as several public and private device-specific instructions (see JTAGC block instructions for a list of supported instructions).
- Sharing of the TAP with other TAP controllers via ACCESS_AUX_x instructions
- Bypass register, boundary scan register, and device identification register
- TAP controller state machine that controls the operation of the data registers, instruction register, and associated circuitry

76.3 Functional description

This section explains the JTAGC functional description.

76.3.1 Modes of operation

The JTAGC block uses a power-on reset indication as its primary reset signals. Several IEEE 1149.1–2001 defined test modes are supported, as well as a bypass mode.

76.3.1.1 Reset

The JTAGC block is placed in reset when:

- Power-on reset is asserted
- TMS input is held high for enough consecutive rising edges of TCK to sequence the TAP controller state machine into the Test-Logic-Reset state

Holding TMS high for five consecutive rising edges of TCK guarantees entry into the Test-Logic-Reset state regardless of the current TAP controller state. Asserting power-on reset results in asynchronous entry into the reset state.

When in reset, the following actions occur:

- The TAP controller is forced into the Test-Logic-Reset state, thereby disabling the test logic and allowing normal operation of the on-chip system logic to continue unhindered.
- The instruction register is loaded with the IDCODE instruction.

76.3.1.2 IEEE 1149.1–2001 defined test modes

The JTAGC block supports several IEEE 1149.1–2001 defined test modes. A test mode is selected by loading the appropriate instruction into the instruction register when the JTAGC is enabled. Supported test instructions include EXTEST, HIGHZ, CLAMP, SAMPLE, and SAMPLE/PRELOAD.

Each instruction defines the set of data register(s) that may operate and interact with the on-chip system logic when the instruction is current. Only one test data register path is enabled to shift data between TDI and TDO for each instruction.

The boundary scan register is enabled for serial access between TDI and TDO when the EXTEST, SAMPLE, or SAMPLE/PRELOAD instructions are active. The single-bit bypass register shift stage is enabled for serial access between TDI and TDO when the following instructions are active:

- BYPASS
- HIGHZ
- CLAMP

The functionality of each test mode is explained in more detail in JTAGC block instructions.

76.3.1.3 Bypass mode

When no test operation is required, the BYPASS instruction can be loaded to place the JTAGC block into bypass mode. When in bypass mode, the single-bit bypass shift register is used to provide a minimum-length serial path to shift data between TDI and TDO.

76.3.2 JTAGC reset configuration

When in reset, the TAP controller is forced into the Test-Logic-Reset state, thus disabling the test logic and allowing normal operation of the on-chip system logic. In addition, the instruction register is loaded with the IDCODE instruction.

76.3.3 IEEE 1149.1-2001 (JTAG) TAP

The JTAGC block uses the IEEE 1149.1-2001 TAP for accessing registers. This port can be shared with other TAP controllers on the MCU. Ownership of the port is determined by the value of the currently loaded instruction. For more detail on TAP sharing via JTAGC instructions, see ACCESS_AUX_x instructions .

Data is shifted between TDI and TDO through the selected register starting with the least significant bit, as illustrated in the following figure. This applies for the instruction register, test data registers, and the bypass register.

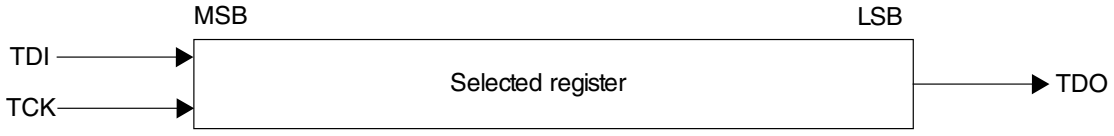
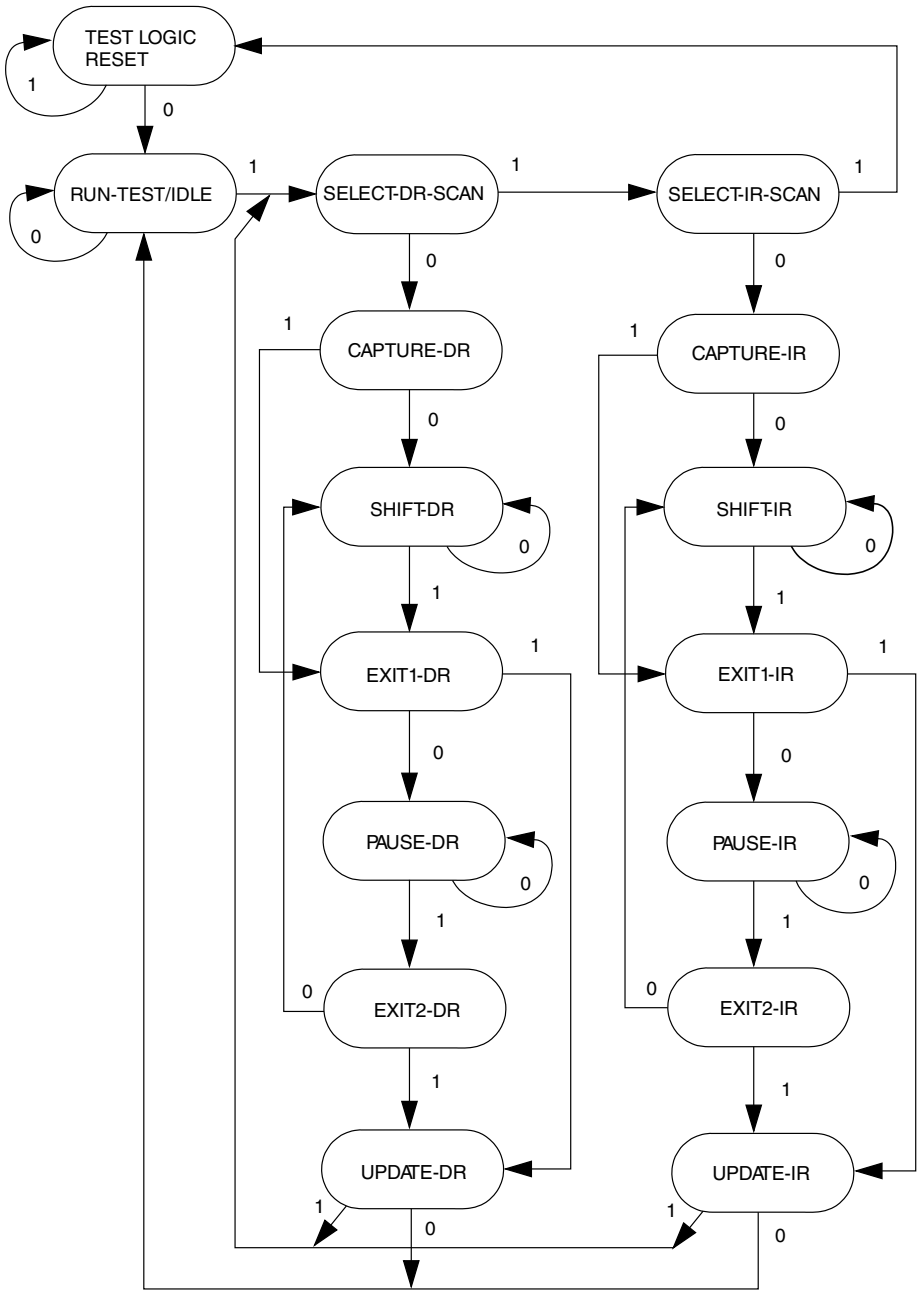


Figure 450. Shifting data through a register

76.3.4 TAP controller state machine

The TAP controller is a synchronous state machine that interprets the sequence of logical values on the TMS pin.

The following figure shows the machine's states. The value shown next to each state is the value of the TMS signal sampled on the rising edge of the TCK signal. As the figure shows, holding TMS at logic 1 when clocking TCK through a sufficient number of rising edges also causes the state machine to enter the Test-Logic-Reset state.



The value shown adjacent to each state transition in this figure represents the value of TMS at the time of a rising edge of TCK.

Figure 451. IEEE 1149.1-2001 TAP controller finite state machine

76.3.4.1 Enabling the TAP controller

The JTAGC TAP controller is enabled by setting the JTAGC enable to a logic 1 value.

76.3.4.2 Selecting an IEEE 1149.1-2001 register

Access to the JTAGC data registers is achieved by loading the instruction register with any of the JTAGC block instructions when the JTAGC is enabled. Instructions are shifted in via the Select-IR-Scan path and loaded in the Update-IR state. At this point, all data register access is performed via the Select-DR-Scan path.

The Select-DR-Scan path is used to read or write the register data by shifting in the data (LSB first) during the Shift-DR state. When reading a register, the register value is loaded into the IEEE 1149.1-2001 shifter during the Capture-DR state. When writing a register, the value is loaded from the IEEE 1149.1-2001 shifter to the register during the Update-DR state. When reading a register, there is no requirement to shift out the entire register contents. Shifting may be terminated after the required number of bits have been acquired.

76.3.5 JTAGC block instructions

The JTAGC implements instructions defined in IEEE 1149.1-2001. See the chip-specific JTAGC information for the JTAGC instructions implemented on this chip. See the IEEE 1149.1-2001 standard for additional information. All undefined opcodes are reserved.

76.3.5.1 IDCODE instruction

IDCODE selects the 32-bit device identification register as the shift path between TDI and TDO. This instruction allows interrogation of the MCU to determine its version number and other part identification data. IDCODE is the instruction placed into the instruction register when the JTAGC block is reset.

76.3.5.2 SAMPLE/PRELOAD instruction

The SAMPLE/PRELOAD instruction has two functions:

- The SAMPLE portion of the instruction obtains a sample of the system data and control signals present at the MCU input pins, and just before the boundary scan register cells at the output pins. This sampling occurs on the rising edge of TCK in the Capture-DR state when the SAMPLE/PRELOAD instruction is active. The sampled data is viewed by shifting it through the boundary scan register to the TDO output during the Shift-DR state. Both the data capture and the shift operation are transparent to system operation.
- The PRELOAD portion of the instruction initializes the boundary scan register cells before selecting the EXTEST or CLAMP instructions to perform boundary scan tests. This is achieved by shifting in initialization data to the boundary scan register during the Shift-DR state. The initialization data is transferred to the parallel outputs of the boundary scan register cells on the falling edge of TCK in the Update-DR state. The data is applied to the external output pins by the EXTEST or CLAMP instruction. System operation is not affected.

76.3.5.3 SAMPLE instruction

The SAMPLE instruction obtains a sample of the system data and control signals present at the MCU input pins, and just before the boundary scan register cells at the output pins. This sampling occurs on the rising edge of TCK in the Capture-DR state when the SAMPLE instruction is active. The sampled data is viewed by shifting it through the boundary scan register to the TDO output during the Shift-DR state. There is no defined action in the Update-DR state. Both the data capture and the shift operation are transparent to system operation.

76.3.5.4 EXTEST external test instruction

EXTEST selects the boundary scan register as the shift path between TDI and TDO. It allows testing of off-chip circuitry and board-level interconnections by driving preloaded data contained in the boundary scan register onto the system output pins. Typically, the preloaded data is loaded into the boundary scan register using the SAMPLE/PRELOAD instruction before the selection of EXTEST. EXTEST asserts the internal system reset for the MCU to force a predictable internal state when performing external boundary scan operations.

76.3.5.5 TEST_LEAKAGE instruction

The TEST_LEAKAGE instruction forces the jtag_leakage output signal to high. It is intended to tristate all output pad buffers and disable all of the part's pad input buffers except JCOMP and TEST. The jtag_leakage signal is asserted at the falling edge of TCK following the TAP controller state machine transition from the Update-IR state to the Run-Test-Idle state. When asserted, the part disables TCK, TMS, and TDI inputs and forces them to a logic 1. The TAP controller state machine remains in the Run-Test-Idle state until the JCOMP input is set to a value other than the JTAGC enable encoding. TEST_LEAKAGE also asserts the internal system reset for the MCU to force a predictable internal state.

76.3.5.6 HIGHZ instruction

HIGHZ selects the bypass register as the shift path between TDI and TDO. When HIGHZ is active all output drivers are placed in an inactive drive state (for example, high impedance). HIGHZ also asserts the internal system reset for the MCU to force a predictable internal state.

76.3.5.7 CLAMP instruction

CLAMP allows the state of signals driven from MCU pins to be determined from the boundary scan register when the bypass register is selected as the serial path between TDI and TDO. CLAMP enhances test efficiency by reducing the overall shift path to a single bit (the bypass register) when conducting an EXTEST type of instruction through the boundary scan register. CLAMP also asserts the internal system reset for the MCU to force a predictable internal state.

76.3.5.8 ACCESS_AUX_x instructions

The JTAGC is configurable to allow other TAP controllers on the device to share the port with it. This is done by providing ACCESS_AUX_x instructions for each of these TAP controllers.

When this instruction is loaded, control of the JTAG pins is transferred to the selected TAP controller. Any data input via TDI and TMS is passed to the selected TAP controller, and any TDO output from the selected TAP controller is sent back to the JTAGC to be output on the pins.

The JTAGC regains control of the JTAG port during the UPDATE-DR state if the PAUSE-DR state was entered. Auxiliary TAP controllers are held in RUN-TEST/IDLE when they are inactive. Instructions not used to access an auxiliary TAP controller on a device are treated like the BYPASS instruction.

76.3.5.9 BYPASS instruction

BYPASS selects the bypass register, creating a single-bit shift register path between TDI and TDO. BYPASS enhances test efficiency by reducing the overall shift path when no test operation of the MCU is required. This allows more rapid movement of test data to and from other components on a board that are required to perform test functions. When the BYPASS instruction is active the system logic operates normally.

76.3.6 Boundary scan

The boundary scan technique allows signals at component boundaries to be controlled and observed through the shift-register stage associated with each pad. Each stage is part of a larger boundary scan register cell, and cells for each pad are interconnected serially to form a shift-register chain around the border of the design. The boundary scan register consists of this shift-register chain, and is connected between TDI and TDO when the EXTEST, SAMPLE, or SAMPLE/PRELOAD instructions are loaded. The shift-register chain contains a serial input and serial output, as well as clock and control signals.

76.4 External signal description

The JTAGC consists of a set of signals that connect to off-chip development tools and allow access to test support functions. The JTAGC signals are outlined in the following table and described in the following sections.

Table 588. JTAG signal properties

Name	I/O	Function	Reset state
TCK	Input	Test clock	Weak pulldown
TDI	Input	Test data in	Weak pullup
TDO	Output	Test data out	High Z[1]
TMS	Input	Test mode select	Weak pullup

1. TDO output buffer enable is negated when the JTAGC is not in the Shift-IR or Shift-DR states. A weak pull may be implemented at the TDO pad for use when JTAGC is inactive.

76.4.1 Test clock input (TCK)

Test Clock Input (TCK) is an input pin used to synchronize the test logic and control register access through the TAP.

76.4.2 Test data input (TDI)

Test Data Input (TDI) is an input pin that receives serial test instructions and data. TDI is sampled on the rising edge of TCK.

76.4.3 Test data output (TDO)

Test Data Output (TDO) is an output pin that transmits serial output for test instructions and data. TDO is tristateable and is actively driven only in the Shift-IR and Shift-DR states of the TAP controller state machine, which is described in TAP controller state machine.

76.4.4 Test mode select (TMS)

Test Mode Select (TMS) is an input pin used to sequence the IEEE 1149.1-2001 test control state machine. TMS is sampled on the rising edge of TCK.

76.5 Initialization/application information

The test logic is a static logic design, and TCK can be stopped in either a high or low state without loss of data. However, the system clock is not synchronized to TCK internally. Any mixed operation using both the test logic and the system functional logic requires external synchronization.

To initialize the JTAGC block and enable access to registers, the following sequence is required:

1. Place the JTAGC in reset through TAP controller state machine transitions controlled by TMS.
2. Load the appropriate instruction for the test or action to be performed.

76.6 Register description

This section provides a detailed description of the JTAGC block registers accessible through the TAP interface, including data registers and the instruction register. Individual bit-level descriptions and reset states of each register are included. These registers are not memory-mapped and can only be accessed through the TAP.

76.6.1 Instruction register

The JTAGC block uses a 8-bit instruction register as shown in the following figure.

The instruction register allows instructions to be loaded into the block to select the test to be performed, or the test data register to be accessed, or both. Instructions are shifted in through TDI when the TAP controller is in the Shift-IR state, and latched on the falling edge of TCK in the Update-IR state. The latched instruction value can only be changed in the Update-IR and Test-Logic-Reset TAP controller states.

Synchronous entry into the Test-Logic-Reset state results in the IDCODE instruction being loaded on the falling edge of TCK. Asynchronous entry into the Test-Logic-Reset state results in asynchronous loading of the IDCODE instruction. During the Capture-IR TAP controller state, the instruction shift register is loaded with the value 0b1, making this value the register's read value when the TAP controller is sequenced into the Shift-IR state.

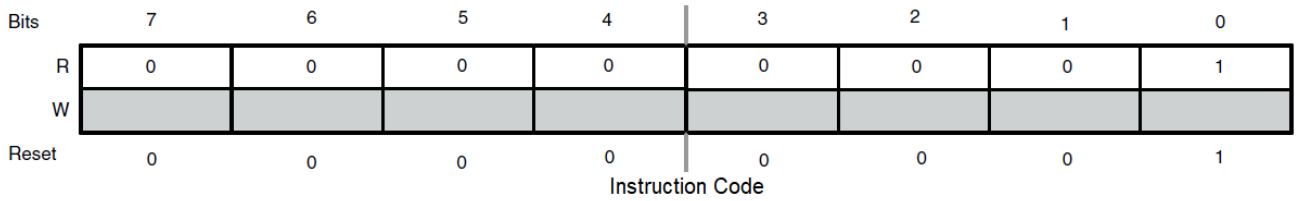


Figure 452. Instruction register

76.6.2 Bypass register

The bypass register is a single-bit shift register path selected for serial data transfer between TDI and TDO when the following instructions are active:

- BYPASS
- HIGHZ
- CLAMP

After entry into the Capture-DR state, the single-bit shift register is set to a logic 0. Therefore, the first bit shifted out after selecting the bypass register is always a logic 0.

76.6.3 Device identification register

The device identification (JTAG ID) register, shown in the following figure, allows the revision number, part number, manufacturer, and design center responsible for the design of the part to be determined through the TAP.

The device identification register is selected for serial data transfer between TDI and TDO when the IDCODE instruction is active. Entry into the Capture-DR state when the device identification register is selected loads the IDCODE into the shift register to be shifted out on TDO in the Shift-DR state. No action occurs in the Update-DR state.

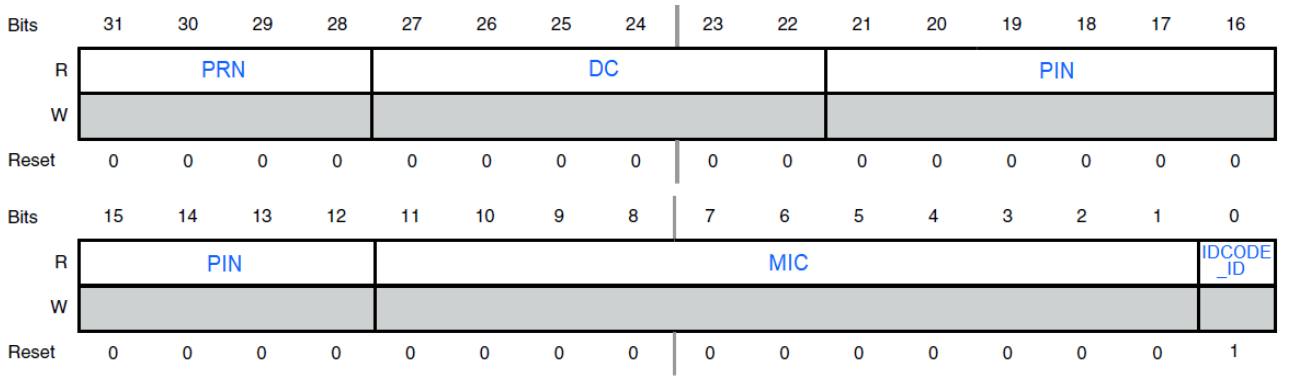


Figure 453. Device identification register

The following table describes the device identification register functions. The device identification register values are described in the chip-specific JTAGC information.

Table 589. Device identification register field descriptions

Field	Function
PRN	Part revision number

Table continues on the next page...

Table 589. Device identification register field descriptions (continued)

Field	Function
	Contains the revision number of the part.
DC	Design center Indicates the design center.
PIN	Part identification number Contains the part number of the device.
MIC	Manufacturer identity code Contains the reduced Joint Electron Device Engineering Council (JEDEC) ID .
IDCODE ID	IDCODE register ID Identifies this register as the device identification register and not the bypass register. Always set to 1.

76.6.4 Boundary scan register

The boundary scan register is connected between TDI and TDO when the EXTEST, SAMPLE, or SAMPLE/PRELOAD instructions are active. It is used to:

- Capture input pin data
- Force fixed values on output pins
- Select a logic value and direction for bidirectional pins

Each bit of the boundary scan register represents a separate boundary scan register cell, as described in the IEEE 1149.1-2001 standard and discussed in Boundary scan. The size of the boundary scan register and bit ordering is device-dependent and can be found in the device BSDL file.

76.7 Glossary

TAP	Test access port
TCK	Test clock
TDI	Test data in
TDO	Test data out
TMS	Test mode select
JCOMP	JTAG compliancy

Chapter 77
JTAG Data Communication (JDC)

77.1 Introduction

The JTAG Data Communication (JDC) module provides the capability to move register data between the IPS and JTAG domains. This facilitates communication between internal resources that access memory-mapped register space and an external tool that accesses the JTAG port.

77.2 Overview

The JDC module consists of:

- IPS-accessible registers
- JTAG-accessible registers
- Associated logic to coordinate movement of data from one register domain to another

The JDC implements the following IPS data registers, occupying separate memory space:

- 1 32-bit memory mapped register that can be read or written via IPS (JTAG Output Data Register (JOUT_IPS)), and whose contents are ported out for capture into a JTAG register (JOUT) to be read via the JTAG port.
- 1 32-bit memory mapped register that can only be read via IPS (JTAG Input Data Register (JIN_IPS)), and whose contents are loaded from a JTAG register (JIN).

JDC indicates when:

- New data has been shifted in via the JTAG port and is ready to be read from the JTAG Input Data Register (JIN_IPS) register
- New data has been written to the JTAG Output Data Register (JOUT_IPS) register and is ready to be read via the JTAG port

A Module Status Register (MSR) captures the state of these flags and also provides the flags to a JTAG register (JOUT) for tool visibility.

There is a single bus interface to port register data out to the JTAGC, and a single bus interface to port data in from the JTAGC. The architecture is shown in the following figure.

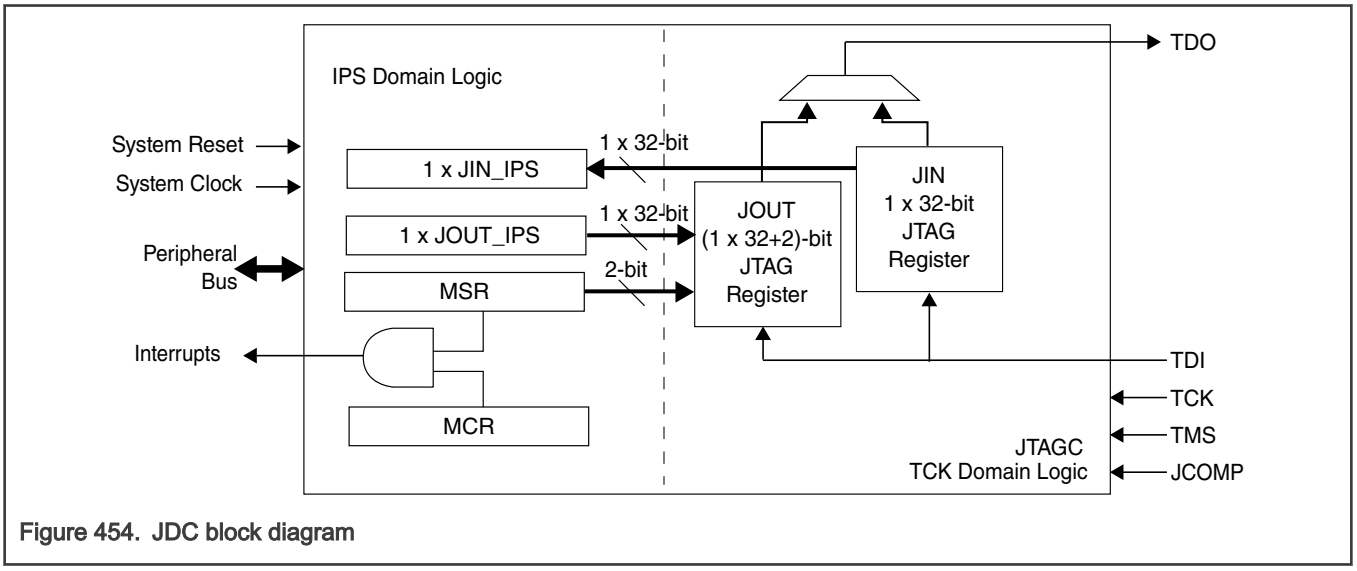


Figure 454. JDC block diagram

77.3 Signal description

JDC signals are listed in the following table.

Table 590. Signal properties

Name	Function	I/O	Reset
TCK	Test clock input	I	—
TMS	Test mode select	I	0
TDI	Test data in	I	0
JCOMP	JTAG compliance pin	I	0
TDO	Test data out	O	HiZ

77.4 Functional description

The JDC module provides the ability to shift in data via the JTAG port and capture that data in memory-mapped register that can be accessed via IPS. It also provides the ability to capture data written to memory-mapped register into a JTAG shift register for output via the JTAG port. An overview of the module functionality is described here.

- Software write to the JOUT_IPS register:
  1. The JDC sets the JOUT_RDY flag bit, indicating new data is available to be read from the JOUT register via the JTAG port.
  2. The MSR reflects the state of the JOUT_RDY bit.
  3. The JDC also captures the state of the JOUT_RDY bit in the JOUT register.
  4. A JTAG read of the JOUT register via execution of the JOUT_READ instruction with a JOUT_RDY bit whose value is logic 1 indicates the register contains new data.
  5. The JDC clears the JOUT_RDY flag bit upon exit of the Capture-DR JTAG state during execution of the JOUT_READ instruction.
  6. Clearing the JOUT_RDY bit indicates to software that a new data value can be written to the JOUT_IPS register.
- JTAG write to the JIN register via execution of the JIN_WRITE JTAG instruction:
  1. The JDC updates the contents of the JIN_IPS register upon exit of the Update-DR state.
  2. An update of the JIN_IPS register sets the JIN_RDY flag bit, indicating new data is available to be read via IPS.
  3. The MSR register reflects the state of the JIN_RDY bit.
  4. The JDC also captures the state of the JIN_RDY bit in the JOUT register.
  5. The JDC clears the JIN_RDY flag bit upon software read of the JIN_IPS register.
  6. A JTAG read of the JOUT register with a JIN_RDY value of logic 0 indicates new data can be written to the JIN register.

77.4.1 JTAG register access

See the JTAGC documentation for information on how to access the JTAG registers.

77.4.1.1 JDC block instructions

The JDC block implements the instructions listed in Table 591. This section gives an overview of each instruction. All undefined opcodes are reserved.

Table 591. JTAG instructions

Instruction	Code[4:0]	Instruction summary
Reserved	00001	Factory debug reserved.
JOUT_READ	00010	Selects JOUT data register. The JDC captures data from JOUT_IPS into JOUT data register upon entry to Capture-DR state when JOUT_READ is active.
JIN_WRITE	01110	Selects JIN data register. The JDC captures data from JIN into JIN_IPS upon exit of Update-DR state when JIN_WRITE is active.
BYPASS[1]	11111	Selects bypass register for data operations.
Reserved	All other opcodes	Decoded to select bypass register.

1. This is an IEEE 1149.1-2001 defined instruction. See the IEEE 1149.1-2001 standard for more details.

77.5 Secure challenge/response connection procedure

A summary of how the JDC can be used to interact with a security module to perform challenge/response password authorization is as follows:

1. After exit of reset, all registers are at their reset values. JIN_RDY bit value of 0 indicates the tool may write data to the JIN register. JOUT_RDY bit value of 0 indicates that software may write data to the JOUT_IPS register.
2. The MCR register is programmed to enable JOUT and JIN interrupts, or not.
3. Once the tool is ready to start the authorization process, it writes a value to the JIN register via execution of the JTAGC JIN_WRITE instruction.
4. Software monitors the state of the JIN_RDY bit or enables the JIN interrupt and uses the interrupt assertion as a trigger to start the authorization process.
5. Once the authorization process is started, software writes the first JOUT value to the JOUT_IPS register. This sets the JOUT_RDY bit value to 1.
6. Following the initial tool write to the JIN register, the tool polls the JOUT register to determine when software has provided a challenge word. Upon reading the JOUT register with JOUT_RDY bit set, the tool records the data value to be used to generate the appropriate response. The read of the JOUT register clears the JOUT_RDY bit. The clearing of the JOUT_RDY bit also sets the JOUT_INT bit and asserts the JOUT interrupt if the MCR[JOUT_IEN] is set.
7. With the JOUT_RDY bit cleared, software may now provide the next challenge word.
8. Once the tool has generated a response word, it uses the value of the JIN_RDY bit from the JOUT register read to determine when it can perform a write to the JIN register to provide the response.
9. The security module can provide additional challenge words and read back the corresponding response words as needed, repeating the process.

Simply providing a password without using the challenge/response protocol can be done in a similar way. The tool provides 32-bits of the password at a time with each write to the JIN register, and monitors the JIN_RDY bit value of the JOUT register to determine when the next JIN register value can be written. There is no need for software to write the JOUT_IPS register in this case.

For details related to the lifecycle stage at which this process is performed, see the "Hardware Security Engine" chapter.

77.6 JDC register descriptions

The following sections describe the implemented 32-bit registers. Only 32-bit accesses are valid. The effects of access that are not 32 bits are not defined.

77.6.1 JDC memory map

JDC base address: 4039_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration Register (MCR)	32	RW	0000_0000h
4h	Module Status Register (MSR)	32	RW	0000_0000h
8h	JTAG Output Data Register (JOUT_IPS)	32	RW	0000_0000h
Ch	JTAG Input Data Register (JIN_IPS)	32	R	0000_0000h

77.6.2 Module Configuration Register (MCR)

Offset

Register	Offset
MCR	0h

Function

The MCR enables the interrupt outputs of the JDC. This register is reset by system destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															JIN_IEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															JOUT_IEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 JIN_IEN	JIN Interrupt Enable.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Setting MSR[JIN_INT] bit does not assert the JIN interrupt 1b - Setting MSR[JIN_INT] bit asserts the JIN interrupt
15-1 —	Reserved
0 JOUT_IEN	JOUT Interrupt Enable. 0b - Setting MSR[JOUT_INT] bit does not assert the JOUT interrupt 1b - Setting MSR[JOUT_INT] bit asserts the JOUT interrupt

77.6.3 Module Status Register (MSR)

Offset

Register	Offset
MSR	4h

Function

The MSR holds the JTAG register status and interrupt bits. This register is reset by system destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													JIN_RDY	Reserved	JIN_INT
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved													JOUT_RDY	Reserved	JOUT_INT
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 JIN_RDY	JIN Ready (read only). 0b - Cleared upon software read of JIN_IPS contents via IPS 1b - Set when new data is written to the JIN_IPS register
17 —	Reserved
16 JIN_INT	JIN Interrupt. 0b - Cleared by writing logic 1 1b - Set when new data is written to the JIN_IPS register
15-3 —	Reserved
2 JOUT_RDY	JOUT Ready (read only). 0b - Cleared upon tool read of JOUT register via JTAG port 1b - Set when new data is written to the JOUT_IPS register
1 —	Reserved
0 JOUT_INT	JOUT Interrupt. 0b - Cleared by writing logic 1 1b - Set when JOUT_RDY bit is cleared by tool reading JOUT register

77.6.4 JTAG Output Data Register (JOUT_IPS)

Offset

Register	Offset
JOUT_IPS	8h

Function

The JOUT_IPS register holds data written via IPS. The JDC captures the JOUT_IPS contents into the JOUT register to be read via the JTAG port. This register is reset by system destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Data															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Data															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 Data	JOUT_IPS data.

77.6.5 JTAG Input Data Register (JIN_IPS)

Offset

Register	Offset
JIN_IPS	Ch

Function

The JIN_IPS register holds data written to the JTAG input data register (JIN) via the JTAG port, where the data can be read via IPS. Any IPS write to the JIN_IPS register returns a transfer error. This register is reset by system destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Data															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Data															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 Data	JIN_IPS data.

77.7 Non-memory-mapped register definition

The JDC also implements two JTAG accessible registers that are not memory-mapped. One JTAG register shifts in data to be placed in the JIN_IPS register. The other JTAG register captures data from the JOUT_IPS register plus ready status from the MSR, to be shifted out via the JTAG port.

77.7.1 JTAG output data register (JOUT)

The JOUT register captures data from the JOUT_IPS register upon execution of the JOUT_READ JTAG instruction. It also holds the JIN_RDY and JOUT_RDY status bits. This register is reset by system destructive reset and JTAG reset. The reset value of the JOUT register is 0. The following figure shows the format of the JOUT register.

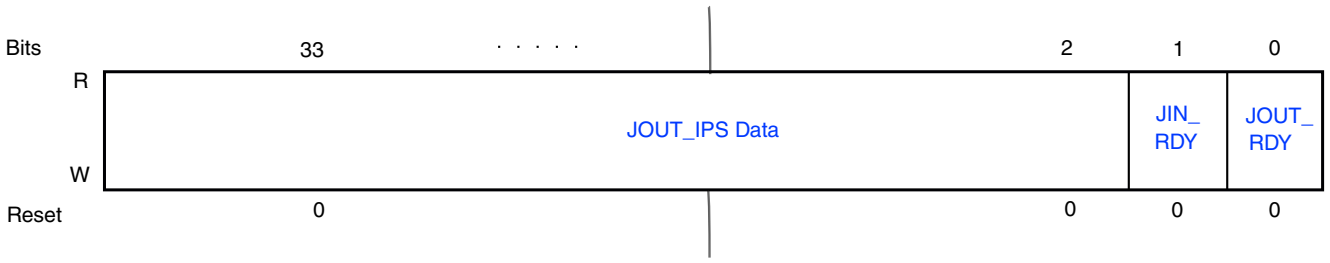


Figure 455. JOUT

The JOUT register is described in the following table.

Table 592. JOUT JTAG register field descriptions

Field	Function
JOUT_IPS Data	JOUT_IPS Data Data value from JOUT_IPS register
JIN_RDY	JIN_RDY State of JIN_RDY bit from MSR
JOUT_RDY	JOUT_RDY State of JOUT_RDY bit from MSR

77.7.2 JTAG input data register (JIN)

The external tool writes data to the JIN register via JTAG during execution of the JIN_WRITE JTAG instruction. The JDC later captures JIN data in the JIN_IPS register to be read via IPS. This register is reset by system destructive reset and JTAG reset. The reset value of the JIN register is 0. The following figure shows the format of the JIN register.

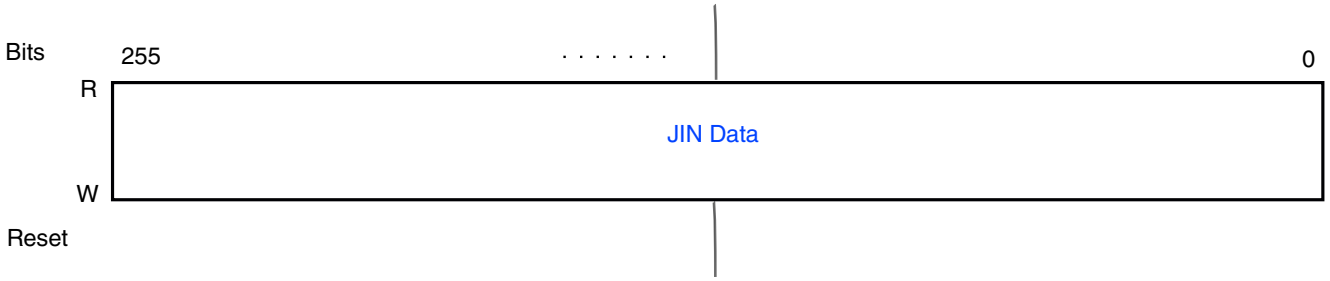


Figure 456. JIN

The JIN register is described in the following table.

Table 593. JIN JTAG register field descriptions

Field	Description
JIN Data	Contains data to be captured in JIN_IPS register upon exit of Update-DR state when executing WRITE_JIN JTAG instruction.

77.8 Glossary

IPS	Internal peripheral system
XBAR	Crossbar bus interface

Chapter 78
Temperature Sensor (TempSense)

78.1 Introduction

The TEMPSENSE module contains control registers for use with the Temperature Sensor hard block. These registers use a peripheral bus interface to communicate with the chip.

78.1.1 Overview

TEMPSENSE is used to measure the temperature on the chip through an ADC.

78.1.2 Block diagram

The functional structure of the TEMPSENSE can be seen in the block diagram in Figure 457.

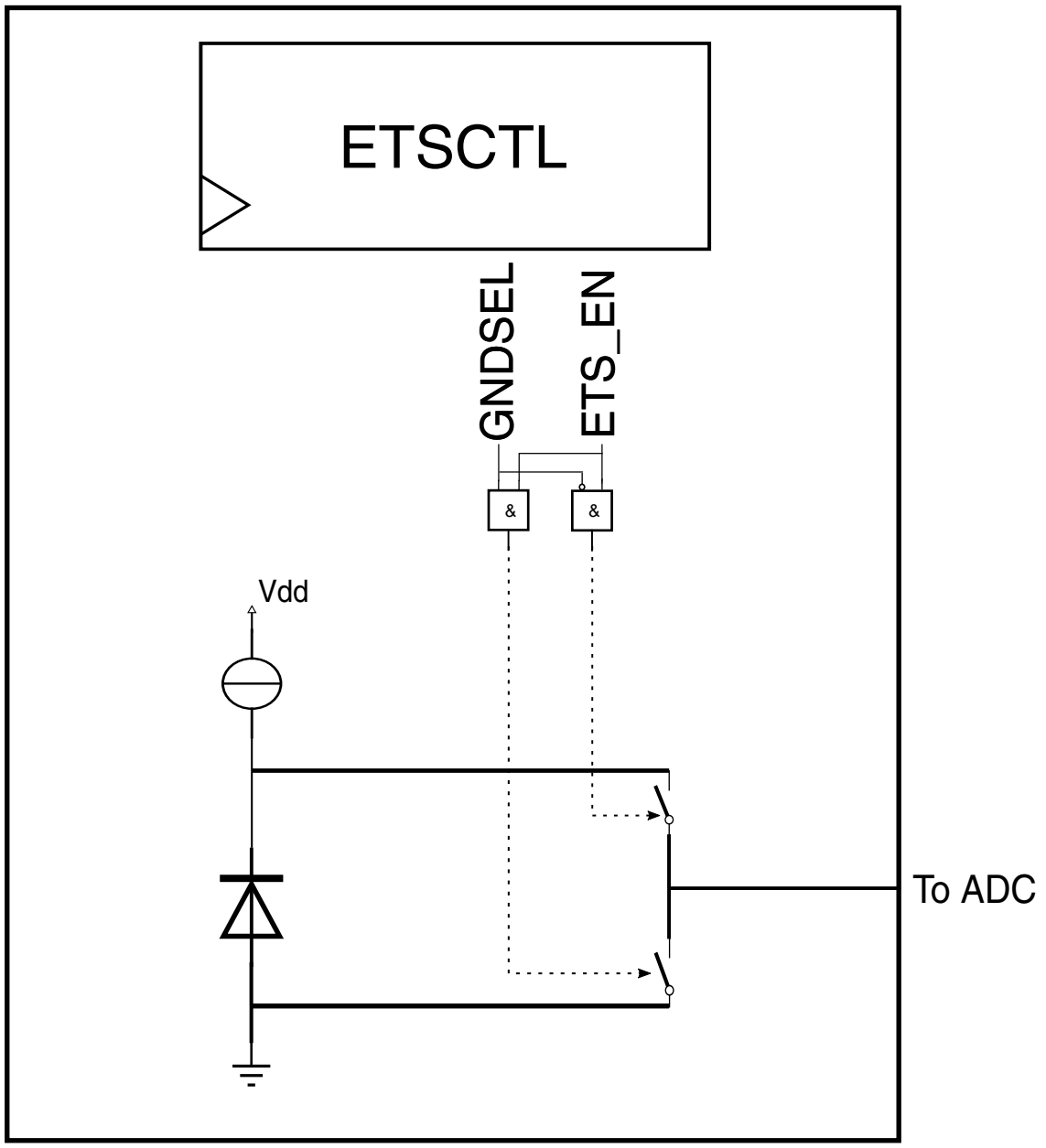


Figure 457. TEMPSENSE block diagram

78.1.3 Features

The TEMPSENSE includes the following features:

- Provide a voltage proportional to the temperature which will be read out by ADC
- Ground selection for improved precision in the ETS. The ETS ground could be different from the ADC ground. By exposing it to the ADC, the temperature calculation could be improved as the voltage value will be more precise.

78.2 Functional description

78.2.1 General

The TEMPSENSE module is designed to measure the temperature on the chip. The TEMPSENSE output can be read by an ADC on demand so that the software can determine the current die temperature. The software should enable the TEMPSENSE by setting ETSCTL[ETS_EN].

Coefficients are used to allow a simple and accurate calculation of linearized temperature directly from the ADC. The increase of the sampling time of the ADC from the minimum sampling value is increasing the temperature accuracy.

78.2.2 Conversion voltage Temperature

Solve the equation for the conversion from voltage to temperature. where V_{ETS} is the difference of V_{be} and V_{GND}. V_{GND} is expose on the ADC output when ETSCTL[GNDSEL] is set.

$$T_{ETS} = TCA_0 + TCA_1 \cdot V_{ETS} + TCA_2 \cdot V_{ETS}^2$$

The coefficients TCA0, TCA1 and TCA2 are read from the corresponding registers. They are stored in a signed fixed-point format as the folowing TCAx(11,4) (1 bit for the sign, 11 bits for the integer part and 4 bits for the decimal part).

The calculation of the temperature should be done with the actual coefficient values provided in the TCAx fields. See the example below for an ambient temperature of 25C, and an ADC reference voltage of 5V.

$$TCA0 = (000110100100.1110)_2 = (000110100100)_2 + (1110)_2 \times \frac{1}{2^4} = +((420)_{10} + (14)_{10} \times \frac{1}{2^4}) = +420.8750\ C$$

$$TCA1 = (100010101111.1110)_2 = (100010101111)_2 + (1110)_2 \times \frac{1}{2^4} = -((175)_{10} + (14)_{10} \times \frac{1}{2^4}) = -175.8750\ C/V$$

$$TCA2 = (100000011101.1110)_2 = (100000011101)_2 + (1110)_2 \times \frac{1}{2^4} = -((29)_{10} + (14)_{10} \times \frac{1}{2^4}) = -29.8750\ C^2/V^2$$

$$V_{ETS} = (010110001011)_2 = (1419)_{10} \times \frac{Vref}{2^{12}} = 1.7322\ V$$

By using the formula the junction temperature calculated is 26.5838 C.

The maximal calculation error is 0.0313 Degrees

If the supply of the ETS module is 3.3V, a conversion is necessary to calculate the accurate temperature.

$$V_{ETS} = (100001100110)_2 = (2150)_{10} \times \frac{Vrefadc}{2^{12}} \times \frac{Vsupply}{Vrefadc} = (2150)_{10} \times \frac{Vsupply}{2^{12}} = 1.7322\ V$$

78.3 Memory map and register definition

78.3.1 Transfer error description

The following actions will cause a transfer error and will not change the register content:

- Any access to an unused register address
- Write access to a read-only register (TCAx)

78.3.2 TempSense register descriptions

78.3.2.1 TempSense memory map

TEMPSENSE base address: 4037_C000h

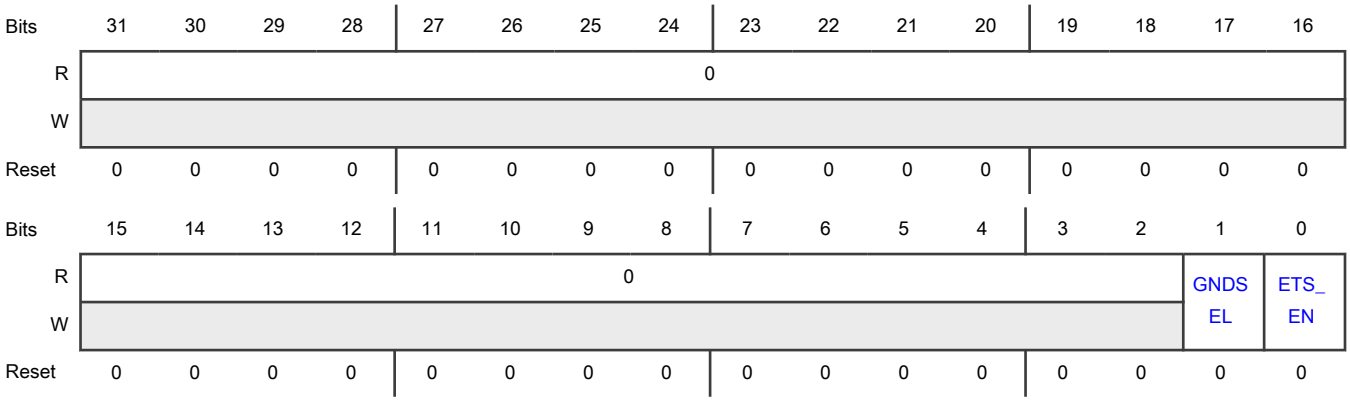
Offset	Register	Width (In bits)	Access	Reset value
0h	ETS Control (ETSCTL)	32	RW	0000_0000h
8h	Temperature Coefficient (TCA0)	32	R	See section
Ch	Temperature Coefficient (TCA1)	32	R	See section
10h	Temperature Coefficient (TCA2)	32	R	See section

78.3.2.2 ETS Control (ETSCTL)

Offset

Register	Offset
ETSCTL	0h

Diagram



Fields

Field	Function
31-2 —	Reserved
1 GNDSEL	Ground selection 0b - No exposure of the ground 1b - Expose ground on the ADC output if ETS_EN=1
0	Temperature Sensor enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
ETS_EN	Power up the ETS. 0b - Power down 1b - Functional mode

78.3.2.3 Temperature Coefficient (TCA0)

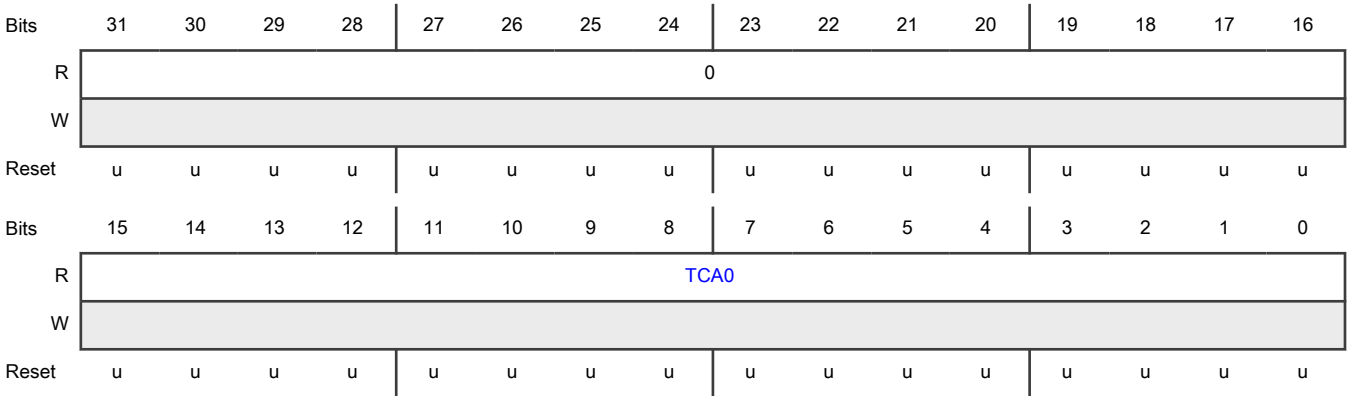
Offset

Register	Offset
TCA0	8h

Function

This register contains the coefficient TCA0 needed to calculate the temperature.the reset value is specific for each chip.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TCA0	Temperature coefficient A0 see Conversion voltage Temperature for the usage of this coefficient

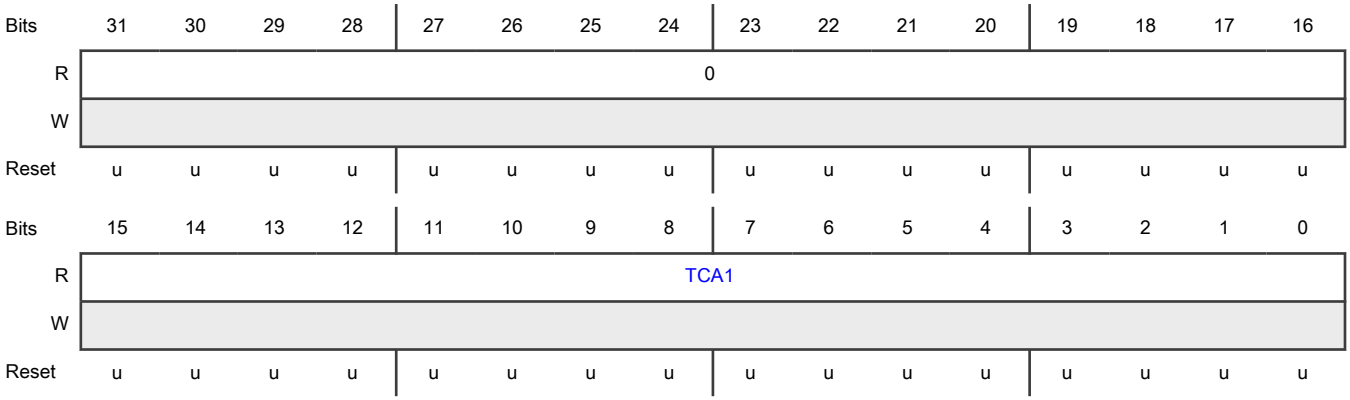
78.3.2.4 Temperature Coefficient (TCA1)

Offset

Register	Offset
TCA1	Ch

Function
This register contains the coefficient TCA1 needed to calculate the temperature.the reset value is specific for each chip.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TCA1	Temperature coefficient A1 see Conversion voltage Temperature for the usage of this coefficient

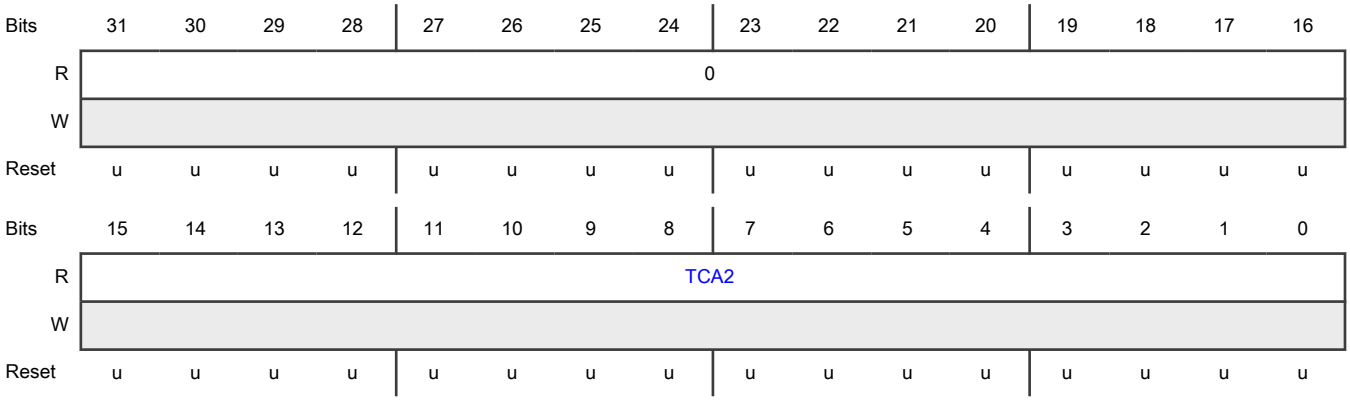
78.3.2.5 Temperature Coefficient (TCA2)

Offset

Register	Offset
TCA2	10h

Function
This register contains the coefficient TCA2 needed to calculate the temperature.the reset value is specific for each chip.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TCA2	Temperature coefficient A2 see Conversion voltage Temperature for the usage of this coefficient

78.4 Glossary

ADC	Analog to digital converter
ETS	Engineering temperature sensor
V_{be}	Base emitter voltage
V_{ETS}	Engineering temperature sensor voltage
V_{GND}	Ground voltage

Appendix A
Revision History

A.1 Revision history

Table 594. Reference manual changes

Revision	Date	Changes
Rev.3	09/2024	• About This Manual — In section Special terms- In table Special terms- Updated the Meaning of "Reserved". — Updated the Figure- Register figure conventions. • Introduction — In section MWCT2xxxS product series- Updated the second bullet from "32-bit Arm Cortex-M7 core with IEEE-754 compliant SPFPU, executing up to 300 MHz" to "32-bit Arm Cortex-M7 core with IEEE-754 compliant SPFPU, executing up to 160 MHz." — In section MWCT2xxxS product series- Added a sub-bullet "Core to core communication" under Stacks/libraries. • Memory Map — In section TCM modes of operation- In table TCM modes of operation- Updated the CM7_n_TCM backdoor enabled from No to Yes for WAIT mode when PRTN2_COFBi_CLKEN[REQ62+n] is 0. • Miscellaneous Control Module (MCM) — In section Introduction- Updated "refer to CM7 overview chapter" to "see the Cortex-M7 core overview chapter". — In the description of Local Memory Descriptor 0 (LMEM_DESC_0) register- Updated the note from "The reserved bits can be readable and writeable (instead of read as zero, write ignored), but if the reserved is set, they do not control anything." to "You can read and write to the reserved bits (instead of read as zero and write ignored). Writing to any of these bits has no functional impact." — In section Interrupts- Changed the bullet from "FPU invalid operation interrupt is enabled (FDZCE) and an invalid operation occurs (FDZC)" to "FPU invalid operation interrupt is enabled (FIOCE) and an invalid operation occurs (FIOC)." — In section Memory map and register descriptions- Updated the Note. — Added Processor Identifier (PID) register. • Miscellaneous System Control Module (MSCM)

Table continues on the next page...

Table 594. Reference manual changes

Revision	Date	Changes
		— In Chip-specific MSCM information- Added section- Reporting of core-to-core interrupts". — In section Memory map and register definition- Deleted the note. — Deleted registers- Interrupt Router CP0 Interrupt Status (IRCP0ISR0), Interrupt Router CP0 Interrupt Generation (IRCP0IGR0), Interrupt Router CP0 Interrupt Status (IRCP0ISR1), Interrupt Router CP0 Interrupt Generation (IRCP0IGR1), Interrupt Router CP0 Interrupt Status (IRCP0ISR2), Interrupt Router CP0 Interrupt Generation (IRCP0IGR2), Interrupt Router CP0 Interrupt Status (IRCP0ISR3), Interrupt Router CP0 Interrupt Generation (IRCP0IGR3), Interrupt Router CP1 Interrupt Status (IRCP1ISR0), Interrupt Router CP1 Interrupt Generation (IRCP1IGR0), Interrupt Router CP1 Interrupt Status (IRCP1ISR1), Interrupt Router CP1 Interrupt Generation (IRCP1IGR1), Interrupt Router CP1 Interrupt Status (IRCP1ISR2), Interrupt Router CP1 Interrupt Generation (IRCP1IGR2), Interrupt Router CP1 Interrupt Status (IRCP1ISR3), and Interrupt Router CP1 Interrupt Generation (IRCP1IGR3). — In the description of CPXTYPE[PERSONALITY]- Changed the processor from CPx = Cortex-M7_1 to CPx = Cortex-M7_3 and changed its Personality value from 43_4D_37_31h to 43_4D_37_33h. — In Enable Interconnect Error Detection (ENEDC) register- Updated the access type of Field 18 and 3 from R/W to ROZ. — Updated the description of CPXCFG0[ICSZ] and CPXCFG0[DCSZ]- Removed cache size information and added a note referencing the MCM chapter for details on cache sizes. • Virtualization Wrapper (VIRT_WRAPPER) — In section VIRT_WRAPPER memory map register descriptions- Updated note from "Access to reserved spaces outside the register bank generates the transfer error. However, access to the reserved spaces within the register bank does not have any effect." to "Access to reserved spaces outside the register bank and holes (unimplemented registers) within register bank generates the transfer error. Access to offset 104h does not generate any transfer error." — In Parameter_n Register (REG_A15_0 - REG_A511_496) and Parameter_n Register (REG_B527_512 - REG_B1023_1008)- Added a note "Any master can access holes (unimplemented registers) of SIUL2 through PDAC0." — Changed from PDAC to PDAC slot as applicable throughout the document.

Table continues on the next page...

Table 594. Reference manual changes

Revision	Date	Changes
		— Updated section Overview. — In section Glossary- Updated the description of PDAC. — Added sections Clocking, Initialization, Application Information, and External signals. — In section Virtualization of input multiplexing control registers- Updated the last paragraph from "As the protection granularity is a single INMUX, any virtualization information associated with this INMUX affects the corresponding IMCR directly. Additionally, any consequence that an illegal access has on the registers implemented within the REG_PROT wrapper is inhibited. For this purpose, any access to the mirrored address space and the corresponding SLBR is also observed." to "As the protection granularity is a single INMUX, any virtualization information associated with this INMUX affects the corresponding IMCR directly." — Updated the section Virtualization of interrupt control registers and Table SIUL2 memory map overview and protection. • System Integration Unit Lite2 (SIUL2) — Updated the SIUL2 block diagram. — Updated the section External interrupt initialization. — In section SIUL2 register descriptions- In the note "For the array of 16-bit registers PGPDO*n* and PGPDI*n*": rephrased the second bullet-item. — Added bit-field values 'Low' and 'High' for PGPDOn and PGPDIn registers. — In the function of register SIUL2 Input Multiplexed Signal Configuration (IMCR0 - IMCR378)- Added 2 bullets ◦ "These registers are a part of SIUL memory map but physical implementation of these register is a part of IOMUX RTL. ◦ SIUL2 interprets accesses to MSCRn at module level." — Added section External signal description, Clocking, and Initialization. — Moved block diagram related information to a new section Block diagram. • Direct Memory Access Multiplexer (DMAMUX) — Updated DMAMUX block diagram. — Updated section Features, Modes of operation, Functional description. — Updated Figure DMAMUX triggered channels.

Table continues on the next page...

Table 594. Reference manual changes

Revision	Date	Changes
		— Added Glossary. • Enhanced Direct Memory Access (eDMA) — In Chip-specific section- Added section eDMA ID replication. — Rename the section Introduction to Overview. — Renamed 'eDMA system block diagram' to 'Block diagram'. — Updated the Figure TCD structure. — Moved 'Modes of operation' under 'Functional description'. — Added missing functions in registers. — Corrected the reset value of TCD registers (SADDR, SOFF, ATTR, SLAST_SDA, DADDR, DOFF, DLAST_SGA, CSR). Changed from 'u' to '0'. — Corrected reset value of TCD registers (NBYTES_MLOFFNO, NBYTES_MLOFFYES, CITER_ELINKNO, CITER_ELINKYES, BITER_ELINKNO, BITER_ELINKYES ,). Changed from 'u' to '0'. — In section Fault reporting and handling- Added statement "If a system bus read or write is terminated with an error, the data transfer is stopped and the appropriate bus error flag set----------If a bus error occurs on the last write prior to switching to the next read sequence, the read sequence executes before the channel terminates due to the destination bus error." Also updated the text "The valid field (VLD) is used to determine if a new error condition exists." to "CHn_ES[ERR] is used to determine if a new error condition exists." — In register Channel System Bus (CH0_SBR - CH31_SBR)- Removed statement "The ATTR register outputs the register values onto the--decoding by the security system" from register Function. — Updated section Channel preemption. — "After the request is completed and the channel is free, the hardware automatically clears the corresponding HRS bit to 0" from HRS bit field in the "Management Page Hardware Request Status Register". — Added topic "External signals". — Added section Interrupts. — In the description of TCD0_CSR - TCD31_CSR[ESDA]- Changed the statement from "The value written to system memory is the last DADDR value prior to the DLAST_SGA offset being applied, or overwritten by an enabled scatter/gather operation" to "The value written to system memory is the next

Table continues on the next page...

Table 594. Reference manual changes

Revision	Date	Changes
		DADDR value prior to the DLAST_SGA offset being applied, or overwritten by an enabled scatter/gather operation". • Interrupt Monitor (INTM) — Updated section Block diagram. — Updated Registers, functional description topics. — Deleted section Glossary. — Added sections Initialization, Clocking, Interrupts, and External signals. • Extended Resource Domain Controller (XRDC) — In Chip-specific section- In section MDAC configuration- Deleted XRDC_MDAC3 instance from Table MDAC configuration. — In section Domain assignment- In Overview- added text "If the value of MDACFGm[NMDAR] is 1, which means a single Ww register for a given master ..... you must be aware of any potential access rights granted for this DID." Also, added a bullet Register settings for DFMT1 direct domain assignment transactions. — In section XRDC transaction examples- Added a bullet "DFMT1 direct domain assignment example-single MDA". — Added section "DFMT1 direct domain assignment example-single MDA" and its sub-sections. — Updated description of MRGD_W3_0 - MRGD_W3_35[VLD] — In MDA_W0_1_DFMT1 - MDA_W0_5_DFMT1[PA] field description Note, changed "secure" to "privileged" — In section MDA register structure- Added a note. — Updated description of (MDA_Wn_m_DFMT0[VLD]) and (MDA_Wn_m_DFMT1[VLD]) — In section Transaction protection- Added a note. • Fast Crystal Oscillator Digital Controller (FXOSC) — Updated section Overview. — Added section Block diagram. — Added sections Clocking, Interrupts, and External signals. — Added field ALC_D in FXOSC Control Register (CTRL). — Renamed section Initialization information to Initialization. Added a note "See Hardware design guide for further details and the recommended circuit for each mode." in 5th bullet. • Reset Generation Module (MC_RGM)

Table continues on the next page...

Table 594. Reference manual changes

Revision	Date	Changes
		— In section Reset state machine- Added a note "See chip-specific MC_RGM information for chip-specific reset sequence details." — In Destructive Event Status Register (DES)- Added a note in the description of field F_POR. — In section Functional reset entry timer- Deleted the text "See the chip-specific section for the default value of timeout value and the details on the chip behavior in case of functional reset entry timer timeout." — In section Functional reset escalation- Added a note "Functional counter increments for each reset source for which escalation is enabled. For details, see "Reset sources" table in the Reset chapter." — In section Destructive reset escalation- Added a note "Destructive counter increments for each reset source for which escalation is enabled. For details, see "Reset sources" table in the Reset chapter." — In section External signal description- In Table MC_RGM external signals- Updated the description of RESET_B signal. — In section RESET_B pin assertion and pin safe state control- Updated the last paragraph from "This feature is intended for use only in conjunction with self-test of the main reset domain. See STCU2 chapter for more details on its usage." to "This feature has to be used only with selftest of the main reset domain." — In section Functional reset entry sequence- Updated the second last paragraph. — In section Functional reset exit sequence- Updated the second bullet from "All the processes that take place during the destructive reset sequence have completed. For details on completed destructive reset sequence, see "Reset process sequence" section in the Reset chapter." to "All the processes that take place during the functional reset sequence have completed. For details, see Reset chapter." — Moved content from "External reset" section to "RESET_B assertion control". — Renamed the "External reset assertion control" section to "RESET_B assertion control". — Updated the description of Functional Reset Escalation Counter Register (FREC). — Updated section Reset sources. • Hardware Security Engine (HSE_B)

Table continues on the next page...

Table 594. Reference manual changes

Revision	Date	Changes
		— Updated section Secure RAM from "Secure RAM refers to RAM area that the HSE subsystem access exclusively. The HSE firmware use it to operate as well as to store a copy of the cryptographic keys in service." to "Secure RAM refers to RAM area that the HSE subsystem access exclusively." — Updated section Secure NVM. — Deleted section HSE interface RAM. — In register Configuration RAM Protected Region (CONFIG_RAMPR)- Updated the bit field description of field HARD_LOCK and SOFT_LOCK.

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

© NXP B.V. 2024. All rights reserved.

For more information, please visit: https://www.nxp.com

Date of release: 09/2024
Document identifier: MWCT2xxxS_RM