版本：V1.4

Nanochap

杭州暖芯迦电子科技有限公司

ENS1A

可编程通用刺激全功能芯片

_数据手册

文档修订记录

序号	版本号	修订日期	修订概述	修订人	审核人	批准人	备注
1	V1.0	2022-11-29	创建文档				
2	V1.1	2023-3-30	第二稿				
3	V1.2	2023-3-31	第三稿添加提示				
4	V1.3	2023-7-7	修改相关参数				
5	V1.4	2023-10-30	修改供电电压范围，加入BOOST选择				

目录

表格一览表

插图目录

1 概述

ENS1A 是一款单芯片刺激装置。它集成了一个强大的 ARM 单片机，集成了电池充电器电路，电源开关，高压升压转换器和高灵活性刺激块。ENS1 可以用于具有最小的芯片外组件的目标应用程序。刺激器块设计用于驱动高达 120mA 的双向刺激电流脉冲。本芯片有 8 个通道的驱动程序可用。 ENS1 生成多种刺激模式来支持各种应用程序。该系统可配置为支持中频物理治疗、常规 TENS、肌肉康复和植入式刺激。

1.1 应用

- 肌肉强化与弱肌康复
- 中频物理治疗
- 大脑深层刺激
- 脊髓刺激
- 人工耳蜗植入

1.2 特性

- 工作温度范围：-40℃至 85℃
- 工作电压范围：3V 至 5V
- 集成式 5~60V 高压升压转换器
- 内核
  - 32-位 ARM Cortex-M0 CPU
  - 频率可达 32MHz
- 内存
  - 32K 字节 MTP 内存
  - 8K 字节 SRAM
- 时钟
  - HSI RC 4-32MHz
  - 高达 32MHz 的外部时钟

- LSI RC 约 32KHz
- 32.768kHz 外部晶振 LSE OSC

■ 低功率运行模式

- 睡眠模式
- 低功率运行模式
- 低功率睡眠模式
- 待机模式

■ 4 个高范围驱动器(8 个电极)(最大 60V)

- 输出电流：33uA~ 67mA，255 步进（最大可输出 120mA 电流，配置方法见 4.3.8 章节）
- 输出单位电流：33uA~264uA，8 级
- 2us~无限宽脉冲
- 高达 250 kHz 正弦/三角形/方形或任意波形
- 可用于 TENS，IFT，EMS

■ 8 个中档驱动器(16 个电极)(最大 60V)

- 输出电流：50uA~52 mA，255 步进
- 输出单位电流：50,72 ~ 204ua(8 步)
- 2us~无限宽脉冲
- 高达 250 kHz 正弦/三角形/方形或任意波形
- 可用于 DBS，SCS

■ 23 通道低频驱动器(24 电极)(最大 60V)

- 8uA~ 2ma 输出电流，255 步 8uA
- 2us~无限宽脉冲
- 高达 250 kHz 正弦/三角形/方形或任意波形
- 可用于人工耳蜗植入

■ 外设模拟电路

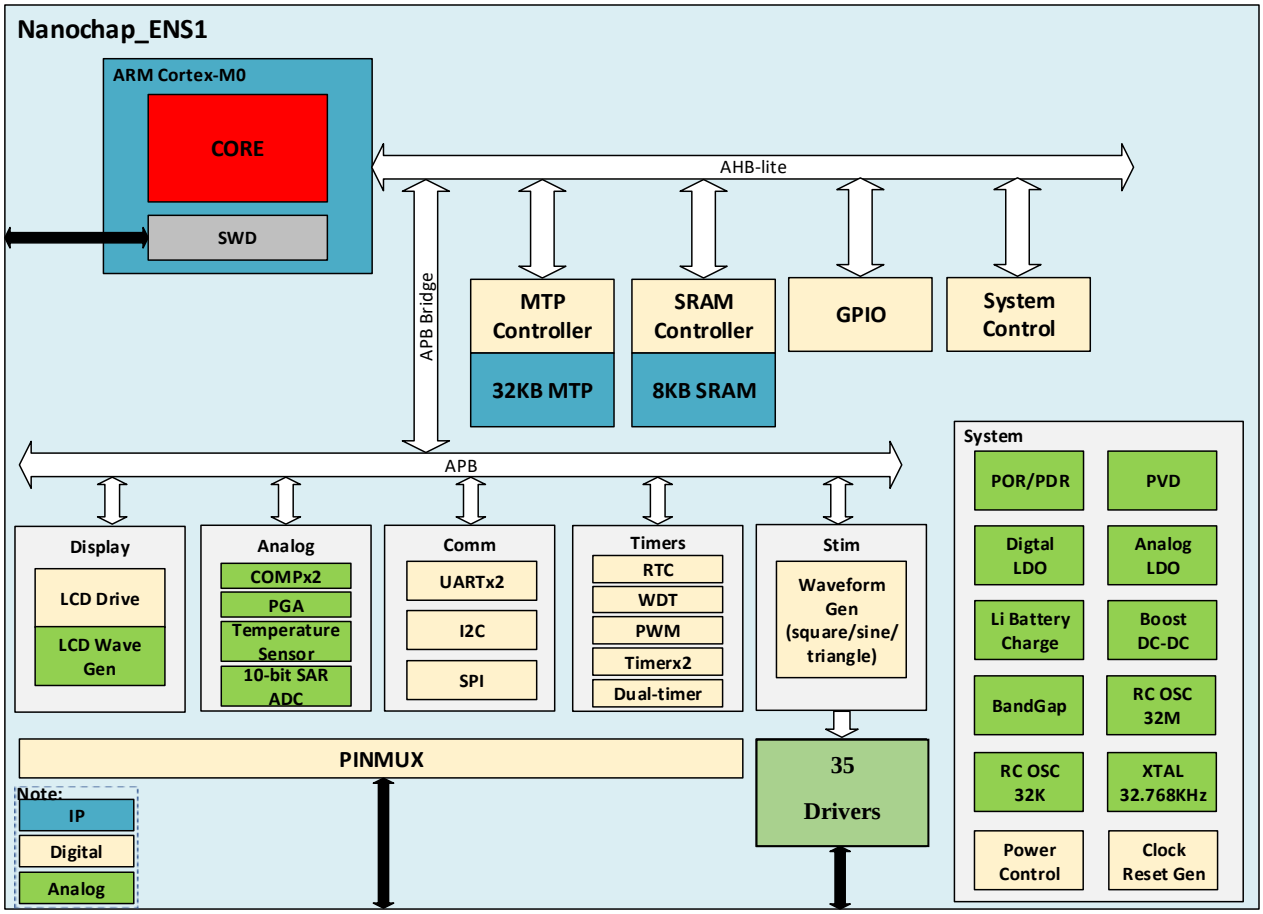
- 12-位 ADC： 0 至 VDD 转换范围
- 温度传感器
- COMPx2
- PGA
- 集成电池充电器
- 通电/断电复位 (POR/PDR)

- 低压检测器 (LVD)

- 24 GPIOs
- LCD 驱动
  - COMx4，SEGx16
- 96-位唯一性标识
- 通信接口
  - UART x2，硬件流控
  - SPI x2，主/从模式
  - I^2C x2，主/从模式
- 各种各样的计时器
  - 实时时钟 (RTC)
  - 看门狗定时器(WDT)
  - 脉宽调制(PWM)
  - 32-位定时器 x2
  - 32-位或 16-位双定时器
  - SysTick 定时器

1.3 框图

图 1 原理框图


Nanochap_ENS1
ARM Cortex-M0
CORE
SWD
AHB-lite
APB Bridge
MTP Controller
32KB MTP
SRAM Controller
8KB SRAM
GPIO
System Control
APB
Display
LCD Drive
LCD Wave Gen
Analog
COMPx2
PGA
Temperature Sensor
10-bit SAR ADC
Comm
UARTx2
I2C
SPI
Timers
RTC
WDT
PWM
Timerx2
Dual-timer
Stim
Waveform Gen (square/sine/ triangle)
PINMUX
35 Drivers
Note:
IP
Digital
Analog
System
POR/PDR
PVD
Digtal LDO
Analog LDO
Li Battery Charge
Boost DC-DC
BandGap
RC OSC 32M
RC OSC 32K
XTAL 32.768KHz
Power Control
Clock Reset Gen

2 引脚及 BOOST 升压电路

2.1 引脚与封装定义

图 2 芯片原理图

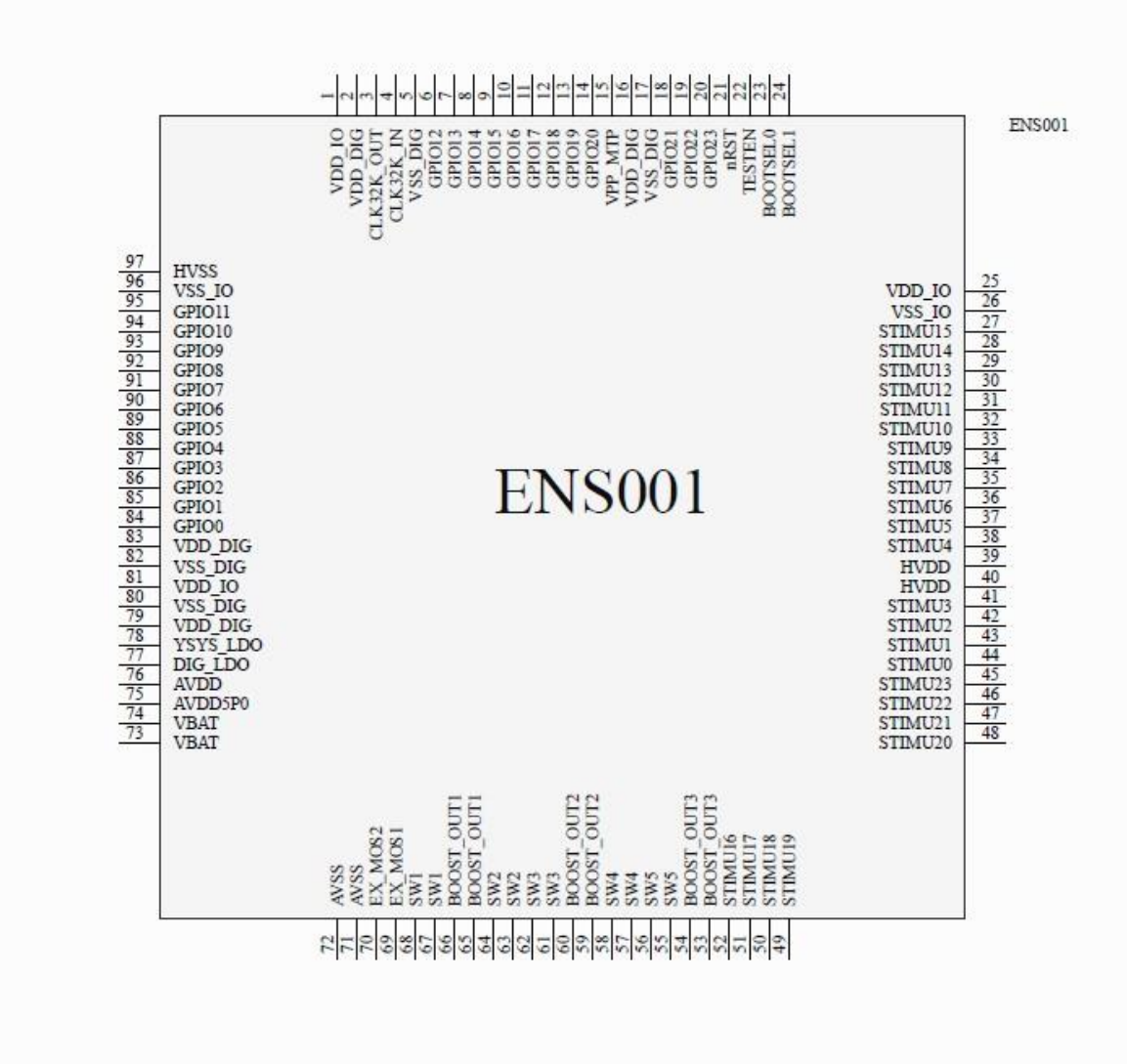


表 1 接口信号表

引脚编号	引脚名称	主要功能(功能 0)	复用功能 1	复用功能 2	复用功能 3	
1	VDD_IO	3.3V 输入输出供应	-	-	-	-
2	VDD_DIG	1.8V 数字核心供应	-	-	-	-
3	CLK32K_OUT	32 kHz 晶体管输出	-	-	-	-
4	CLK32K_IN	32 kHz 晶体管输入	-	-	-	-
5	VSS_DIG	数字接地端	-	-	-	-
6	GPIO12	GPIO12	UART1_RXD	MCO	SEG4	COMP1_VIP0
7	GPIO13	GPIO13	UART1_TXD	SPI0_NSS1	SEG5	COMP1_VIP1
8	GPIO14	GPIO14	UART1_RTS_N	SPI0_NSS2	SEG6	COMP1_VIN0
9	GPIO15	GPIO15	UART1_CTS_N	SPI0_NSS3	SEG7	COMP1_VIN1
10	GPIO16	GPIO16	PWM1_OUT	SPI1_SCK	SEG8	PGA_VIP0
11	GPIO17	GPIO17	PWM2_OUT	SPI1_MOSI	SEG9	PGA_VIP1
12	GPIO18	GPIO18	PWM3_OUT	SPI1_MISO	SEG10	PGA_VIN0
13	GPIO19	GPIO19	PWM4_OUT	SPI1_NSS0	SEG11	PGA_VIN1
14	GPIO20	GPIO20	PWM5_OUT	COMP0_OUT	SEG12	PGA_EXVCM
15	VPP_MTP	MTP 内存 VPP 引脚	-	-	-	-
16	VDD_DIG	1.8V 数字核心供应	-	-	-	-
17	VSS_DIG	数字接地端	-	-	-	-
18	GPIO21	GPIO21	PWM6_OUT	COMP1_OUT	SEG13	ADC_IN0
19	GPIO22	GPIO22	TIMER0_EXTIN	-	SEG14	ADC_IN1
20	GPIO23	GPIO23	TIMER1_EXTIN	-	SEG15	ADC_IN2
21	nRST	复位，低电平，默认 = 1	-	-	-	-
22	TESTEN	TEST 启用，高电平，默认 = 0	-	-	-	-
23	BOOTSEL0	启动选项 0，默认 = 0	-	-	-	-
24	BOOTSEL1	启动选项 1，默认 = 0	-	-	-	-
25	VDD_IO	3.3V 输入输出供应	-	-	-	-
26	VSS_IO	输入输出接地	-	-	-	-
27	STIMU15	驱动 B 和驱动 C 输出/DB_ELE15/DC_ELE7	-	-	-	-
28	STIMU14	驱动 B 和驱动 C 输出/DB_ELE14/DC_ELE6	-	-	-	-
29	STIMU13	驱动 B 和驱动 C 输出/DB_ELE13/DC_ELE5	-	-	-	-
30	STIMU12	驱动 B 和驱动 C 输出/DB_ELE12/DC_ELE4	-	-	-	-
31	STIMU11	驱动 B 和驱动 C 输出/DB_ELE11/DC_ELE3	-	-	-	-

32	STIMU10	驱动 B 和驱动 C 输出/DB_ELE10/DC_ELE2	-	-	-	-
33	STIMU9	驱动 B 和驱动 C 输出/DB_ELE9/DC_ELE1	-	-	-	-
34	STIMU8	驱动 B 和驱动 C 输出/DB_ELE8/DC_ELE0	-	-	-	-
35	STIMU7	驱动 A 和驱动 B 输出/DA_CH3_阳极/DB_ELE7	-	-	-	-
36	STIMU6	驱动 A 和驱动 B 输出/DA_CH3_阴极/DB_ELE6	-	-	-	-
37	STIMU5	驱动 A 和驱动 B 输出/DA_CH2_阳极/DB_ELE5	-	-	-	-
38	STIMU4	驱动 A 和驱动 B 输出/DA_CH2_阴极/DB_ELE4	-	-	-	-
39	HVDD	用于刺激驱动的高压电源	-	-	-	-
40	HVDD	用于刺激驱动的高压电源	-	-	-	-
41	STIMU3	驱动 A 和驱动 B 输出/DA_CH1_阳极//DB_ELE3	-	-	-	-
42	STIMU2	驱动 A 和驱动 B 输出/DA_CH1_阴极/DB_ELE2	-	-	-	-
43	STIMU1	驱动 A 和驱动 B 输出/DA_CH0_阳极/DB_ELE1	-	-	-	-
44	STIMU0	驱动 A 和驱动 B 输出/DA_CH0_阴极/DB_ELE0	-	-	-	-
45	STIMU23	驱动 B 和驱动 C 输出/DB_ELE23/DC_SW7	-	-	-	-
46	STIMU22	驱动 B 和驱动 C 输出/DB_ELE22/DC_SW6	-	-	-	-
47	STIMU21	驱动 B 和驱动 C 输出/DB_ELE21/DC_SW5	-	-	-	-
48	STIMU20	驱动 B 和驱动 C 输出/DB_ELE20/DC_SW4	-	-	-	-
49	STIMU19	驱动 B 和驱动 C 输出/DB_ELE19/DC_SW3	-	-	-	-
50	STIMU18	驱动 B 和驱动 C 输出/DB_ELE18/DC_SW2	-	-	-	-
51	STIMU17	驱动 B 和驱动 C 输出/DB_ELE17/DC_SW1	-	-	-	-
52	STIMU16	驱动 B 和驱动 C 输出/DB_ELE16/DC_SW0	-	-	-	-
53	BOOST_OUT 3	直流-直流升压输出 3	-	-	-	-
54	BOOST_OUT 3	直流-直流升压输出 3	-	-	-	-
55	SW5	直流-直流升压电感器-二极管 5	-	-	-	-
56	SW5	直流-直流升压电感器-二极	-	-	-	-

		管 5				
57	SW4	直流-直流升压电感器-二极管 4	-	-	-	-
58	SW4	直流-直流升压电感器-二极管 4	-	-	-	-
59	BOOST_OUT 2	直流-直流升压输出 2	-	-	-	-
60	BOOST_OUT 2	直流-直流升压输出 2	-	-	-	-
61	SW3	直流-直流升压电感器-二极管 3	-	-	-	-
62	SW3	直流-直流升压电感器-二极管 2	-	-	-	-
63	SW2	直流-直流升压电感器-二极管 2	-	-	-	-
64	SW2	直流-直流升压电感器-二极管 2	-	-	-	-
65	BOOST_OUT 1	直流-直流升压输出 1	-	-	-	-
66	BOOST_OUT 1	直流-直流升压输出 1	-	-	-	-
67	SW1	直流-直流升压电感器-二极管 1	-	-	-	-
68	SW1	直流-直流升压电感器-二极管 1	-	-	-	-
69	EX_MOS1	直流-直流升压器外部主开关 NMOS 的门控信号	-	-	-	-
70	EX_MOS2	直流-直流升压器外部过电压放电 NMOS 的门控信号	-	-	-	-
71	AVSS	模拟地	-	-	-	-
72	AVSS	模拟地	-	-	-	-
73	VBAT	4.2V 电池输入	-	-	-	-
74	VBAT	4.2V 电池输入	-	-	-	-
75	AVDD5P0	5V 直流输入，为电池充电	-	-	-	-
76	AVDD	3.3V LDO 输出	-	-	-	-
77	DIG_LDO	1.8V LDO 输出	-	-	-	-
78	VSYS_LDO	电池充电电路 4.23V LDO 输出	-	-	-	-
79	VDD_DIG	1.8V 数字核心供应	-	-	-	-
80	VSS_DIG	数字接地端	-	-	-	-
81	VDD_IO	3.3V 输入输出供应	-	-	-	-
82	VSS_DIG	数字接地端	-	-	-	-
83	VDD_DIG	1.8V 数字核心供应	-	-	-	-
84	GPIO0	SWCLK	GPIO0	HSE_CLK	-	-
85	GPIO1	SWDIO	GPIO1	-	-	-

86	GPIO2	GPIO2	UART0_RXD	SPI1_NSS1	-	-
87	GPIO3	GPIO3	UART0_TXD	SPI1_NSS2	-	-
88	GPIO4	GPIO4	UART0_RTS_N	SPI1_NSS3	COM0	PGA_OUT
89	GPIO5	GPIO5	UART0_CTS_N	-	COM1	ANA_BIST
90	GPIO6	GPIO6	I2C0_SCL	-	COM2	-
91	GPIO7	GPIO7	I2C0_SDA	-	COM3	-
92	GPIO8	GPIO8	SPI0_SCK	I2C1_SCL	SEG0	COMP0_VIP0
93	GPIO9	GPIO9	SPI0_MOSI	I2C1_SDA	SEG1	COMP0_VIP1
94	GPIO10	GPIO10	SPI0_MISO	—	SEG2	COMP0_VIN0
95	GPIO11	GPIO11	SPI0_NSS0	RTC_1HZ	SEG3	COMP0_VIN1
96	VSS_IO	输入输出接地	-	-	-	-
97	HVSS	高电压接地-接地	-	-	-	-

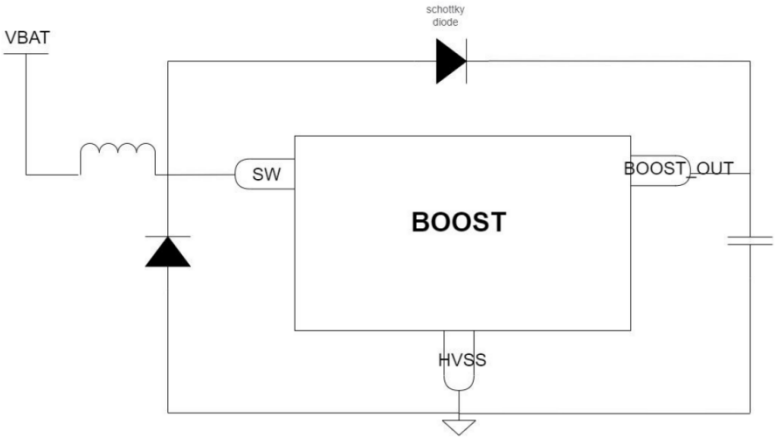
注：设计请参考原理图。如果您需要使用升压电路过压、过温功能，请联系制造商。

2.2 BOOST 升压电路选择

ENS001 芯片自身提供两种升压电路供应用端选择：内置 MOS 升压电路和外置 MOS 升压电路，以上两种 ENS001 升压电路均为芯片的可选配置。

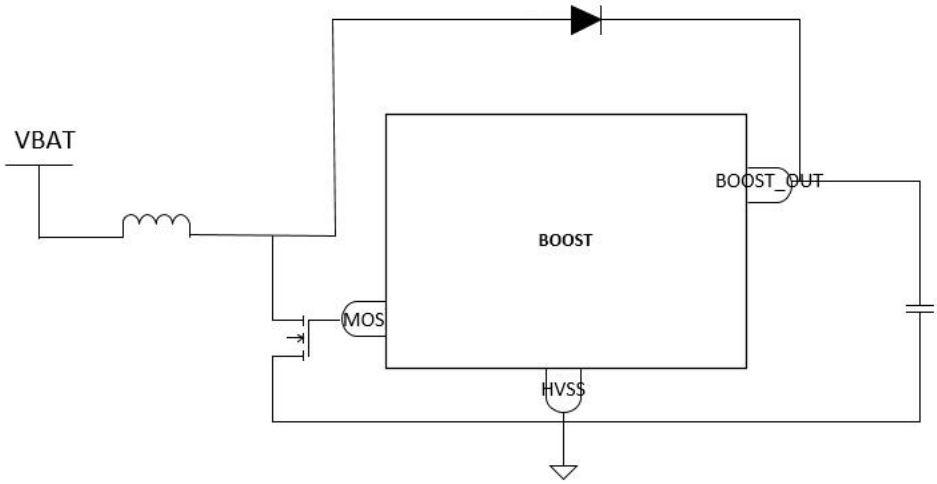
使用 ENS001 芯片内部 MOS 升压电路，芯片外部需接入升压电感、肖特基二极管、滤波电容，使用寄存器配置程序的方式配置内部 SW 信号的频率及占空比，调整 BOOST 电路输出的电压。

图 3 内置 MOS 升压电路图



使用 ENS001 芯片外部 MOS 升压电路，芯片外部需接入升压电感、、外置 MOS 管、肖特基二极管、滤波电容，使用寄存器配置程序的方式配置内部 MOS 控制信号的频率及占空比，调整 BOOST 电路输出的电压。

图 4 外置 MOS 升压电路图



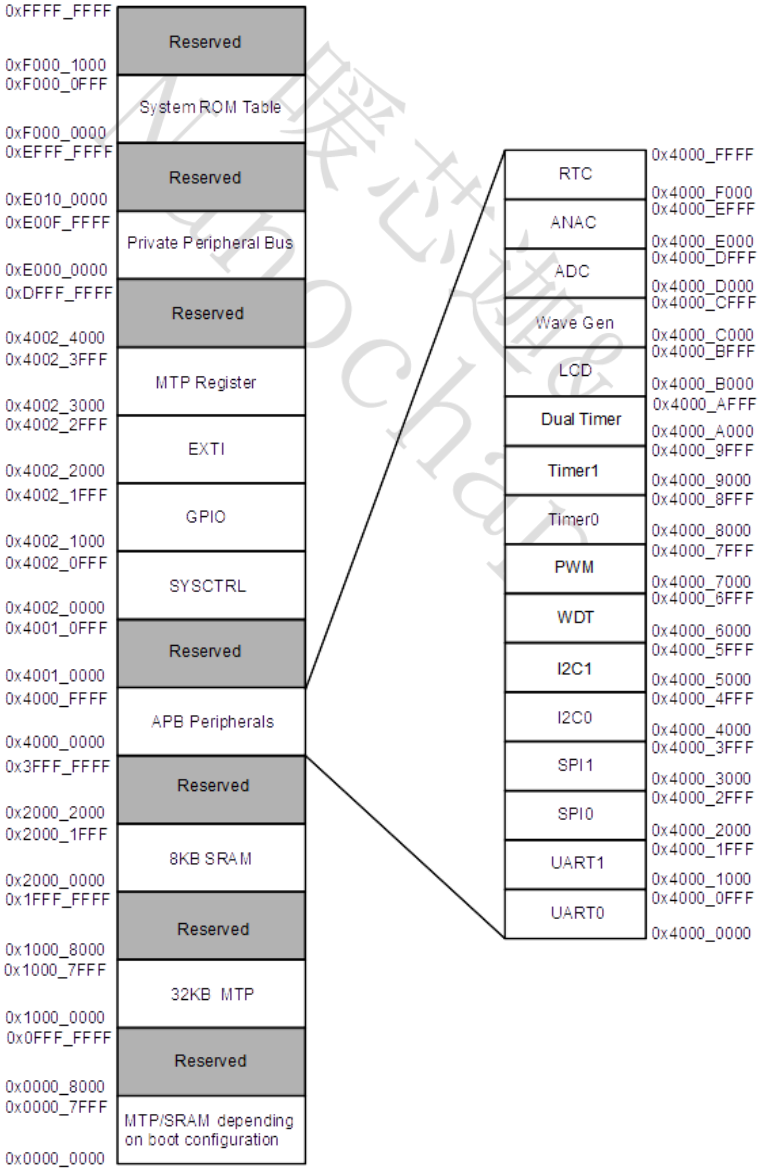
用户端如需要更高输出功率的应用，可使用外加 BOOST 电路方案搭配 ENS001 波形发生刺激功能。

3 内存

3.1 内存映射

程序存储器、数据存储器、寄存器和输入输出端口被组织在相同的线性 4G 字节地址空间中。字节在内存中以小端格式编码。所有没有分配到芯片内存和外设的内存映射区域都被认为是“预留的”。

图 5 内存映射图



3.2 嵌入式 SRAM

嵌入式 SRAM 容量为 8K 字节。可以通过字节、半字(16 位)或全字(32 位)来访问 SRAM。可以在没有等待状态的情况下以最大系统时钟频率寻址这个内存。

SRAM 起始地址为 0x2000_0000。

当通过 BOOT pin 引脚或 REMAP 寄存器选择物理重映射时，CPU 可以从地址 0x0000_0000 访问 SRAM。

3.3 启动配置

通过 BOOT0 和 BOOT1 引脚可以选择三种启动方式，如下表所示。

表 2 启动模式

启动模式选择		启动模式	别名使用
引导 1 引脚	引导 0 引脚		
X	0	MTP 基地址	选择来自基本区域的 MTP 主区域作为应用程序代码使用的引导区域
0	1	MTP 高 4K 字节	MTP 高 4K 字节区域被选为引导加载程序使用的引导区域
1	1	嵌入式 SRAM	嵌入式 SRAM 被选择为用于调试的引导区域

启动延迟结束后，CPU 从地址 0x0000_0000 获取堆栈顶部值,然后从引导内存 0x0000_0004 开始执行代码。

从基地地址的 MTP 主区域引导： MTP 内存在引导内存空间（0x0000_0000）中被别名，但仍然可以从其原始内存空间（0x1000_0000）访问。换句话说，可以从地址 0x0000_0000 或 0x1000_0000 开始访问 MTP 内存内容。

从 MTP 高 4K 字节区引导： MTP 高 4K 字节区域用于引导加载程序，它在引导内存空间(0x0000_0000)中被别名化，但仍然可以从其原始内存空间(0x1000_7000)中访问。剩余的 28K 字节空间用于应用程序代码。

从嵌入式 SRAM 引导： SRAM 在引导内存空间(0x0000_0000)中有别名，但仍然可以从其原始内存空间(0x2000_0000)中访问它。

一旦选择了 BOOT0 和 BOOT1 引脚，应用软件就可以在代码区修改内存。这种修改是通过在 SYSCTRL 寄存器中编程 REMAP 重新映射位来执行的。

BOOT0 和 BOOT1 默认是弱下拉，当从引导加载程序空间引导，即引导 0pin 需要拉高，软件可以清除 GPIO_PD[24]寄存器以节省电力。

4 多次编程存储器（MTP）

4.1 概述

YEG8K32F18B5AA1 是从 Globalfoundries 0.18um BCDlite1.8V–ULL 6V_40V-65V 进程中编出来的一个嵌入式 MTP IP 宏。它的每个存储器阵列均被分成三个内存块，一个 MTP 块（8K x 32 位），一个信息块（32 x 32 位），和一个 EEPROM 块（512x8 位）。

YEG8K32F18B5AA1 支持两种运行模式：用户模式和测试模式。

- 用户模式支持五种内存操作：复位（RESET）/待机（STANDBY）/静态（STATIC）/读取（READ）/内部高电压写入（INTHV WRITE）。
- 测试模式支持四种内存操作；测量存储单元电流（CLEN）/数据保持测试读取（MRGN READ）/数据保持测试内部高电压写入（MRGN INTHV WRITE）/外部高电压对所有 MTP 密度写入（EXTHV WRITE ALL）。

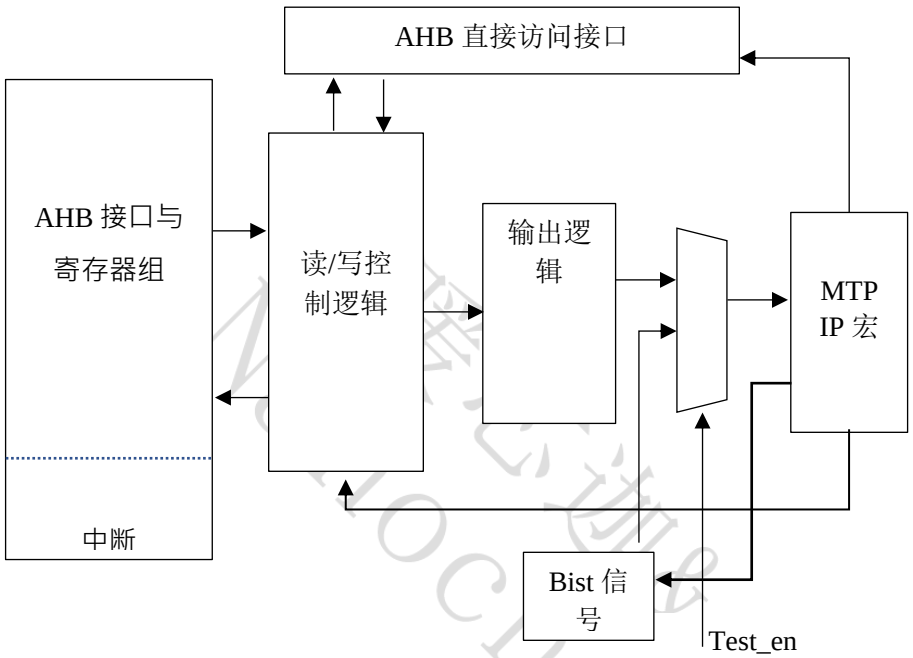
4.1.1 功能列表

- 存储器组成：MTP 块（8K x 32 位），信息块（32 x 32 位），EEPROM 块（512x8 位）。
- MTP 块支持 32 位读取和扇区写入操作。
- 信息块支持 32 位读取和 32 位写入操作。
- EEPROM 块支持字节读取和字节写入操作。
- 数据重新写入，不进行擦除操作。
- 写入时间为非固定值。
- BUSY 信号提供了一种硬件方法，可检测写入操作是否已完成。
- 写入 MTP 密钥：0x5A5A5A5A。

4.2 框图

MTP 控制器的功能框图如下图所示。

图 6 MTP 框图



4.3 功能描述

4.3.1 支持的用户操作方式

用户模式可以在以下五种模式中选择一种

- 复位(RESET)
- 待机(STANDBY)
- 静态(STATIC)
- 读取(READ)
- 内部高电压写入(INTHV WRITE)

工作模式选择如下图所示

表 3 用户操作模式

引脚名称	复位	待机	静态	读取		内部高电压写入	
RESETB2	L	H	H	H		H	
CS	L	L	H	H		H	
READ	L 或 H	L 或 H	L	H		L	
WR	L 或 H	L 或 H	L	L		H	
CLEN	L 或 H	L 或 H	L	L		L	
EEPROM	L 或 H	L 或 H	L 或 H	L 或 H	L	L 或 H	L
SRL	L 或 H	L 或 H	L	L		L	
MRGN	L 或 H	L 或 H	L	L		L	
HVEN	L 或 H	L 或 H	L	L		L	
WRALL	L 或 H	L 或 H	L	L		L	
VPP	VDD2						

4.3.2 在 MTP 块中的扇区写入操作的定义

MTP 块中的存储器阵列由多个扇区集成而成。每个扇区的地址定义和描述如下表。

MTP 块只允许“扇区写入”操作。

不允许将数据随机写入存储器阵列。从目标扇区的第一个地址到最后一个地址连续发起写入操作。

表 4 MTP 块存储器阵列配置

扇区编号	使用地址	扇区编号	使用地址
扇区 0	0000H-03FFH	扇区 4	1000H-03FFH
扇区 1	0400H-07FFH	扇区 5	1400H-17FFH
扇区 2	0800H-0BFFH	扇区 6	1800H-1BFFH
扇区 3	0C00H-0FFFH	扇区 7	1C00H-1FFFH

4.3.3 MTP 块中引导加载扇区的定义

引导加载扇区地址为 MTP 块中的扇区 7。当 DBGPA_EN 和 BOOTLD_WEN 关闭时，引导加载扇区可以由 SWD 写入。该软件应用程序只能读取引导加载扇区。

4.3.4 信息块中的 trim 和 ID 配置信息区域的定义

Trim 配置信息区域地址为 0x0018 ~ 0x001B，设备 ID 配置信息区域地址为 0x001C ~ 0x001E。它们用于存储模拟 trim 值和设备 ID 值，这些值将在上电和复位后自动加载到相关配置寄存器中。

4.3.5 信息块中的用户配置信息区域的定义

用户配置信息区域地址为 0x001F，它用于存储用户配置信息，这些信息将在上电和复位后自动加载到配置寄存器中。

表 5 用户配置信息定义

位	字段名	字段描述	默认
31：24	-	-	-
23：16	BOOTLD_SWDPA_EN	引导加载程序 SWD 访问保护启用： 0x55： 开 其他：关	0x55
15：8	DBGPA_EN	调试端口访问保护启用： 0xAA： 开 其他：关	0xAA
7：0	SEC_ACL_EN	应用程序代码锁启用： 0X33： 启用时钟 其他：禁止时钟	0x33

4.3.6 MTP 用户访问权限

表 6 MTP 访问权限

MTP 区		MTP_DBGPA_EN	SEC_ACL	BOOTLD_SWDPA_EN	SWD	应用
MTP 块	扇区 0-扇区 6	1	0	x	-	读取/写入
			1	x	-	-
		0	0	x	读取/写入	读取/写入
			1	x	读取/写入	-
	引导加载区扇区 7	1	x	1	-	读取
				0	-	读取
		0	x	1	-	读取
				0	读取/写入	读取
信息块	其他区	1	x	x	-	读取/写入
		0	x	x	读取/写入	读取/写入
	Trim 和 ID 区	1	x	x	-	读取
		0	x	x	读取/写入	读取
	用户区	1	x	x	-	读取
		0	x	x	读取/写入	读取
EEPROM 块		1	x	x	-	读取/写入
		0	x	x	读取/写入	读取/写入

4.3.7 时序参数

参数		符号	分钟	类型	最大	单位
VDD1 设置时间		Tpws	100			ns
VDD1 保持时间		Tpwh	100			ns
VDD2 设置时间		Trsts	100			ns
VDD2 保持时间		Trsth	100			ns
RESETB 到 CS 的设置时间		Trscs	40			us
RESETB 到 CS 的保持时间		Trsch	100			ns
CS 至读取/写入设置时间		Tcsctrls	100			ns
地址/数据设置时间		Tads	20			ns
地址持续时间		Tadhr	50			ns
读取脉宽	MTP 和信息块	Trpw	60			ns
	EEPROM 块		250			ns
读取访问时间	MTP 和信息块	Trac			60	
	EEPROM 块				250	
读取周期时间	MTP 和信息块	Trc	80			ns
	EEPROM 块		270			ns
DOUT 持续时间		Tdoh	3			ns
WR 脉宽		Twpw	100			ns
写入的地址/数据持续时间		Tadhw	50			ns
写入时间（变更代码）	MTP 和信息块	Twr		0.5		ms
	EEPROM 块			0.15		ms
写入时间（非变更代码）			1			Us
对 WR 的 BUSY 访问时间		Tbas			100	ns
写入恢复时间		Twrc	100			ns
测量电流等待时间		Tmcw	100			ns
CLEN 地址持续时间		Tadhc	20			ns

4.3.8 MTP 单元电流配置

通过配置 MTP BG Trim register (BG_I_LC, BG_I_MC)数值可用于调整 ENS001 内置单元电流，具体寄存器地址及配置如下。

Bit	Field Name	Attribut	Default	Field Description
31:8	-	RO	0	Reserved
7:4	BG_I_LC	RW	0x8	Analog trim
3:0	BG_I_MC	RW	0x8	Analog trim

通过调整 BG_I_LC, BG_I_MC 对应数值，可相应修改电流源单位电流系数，用于减少或增加单位刺激电流，具体对应关系如下表所示。

BG_I_MC<3:0>	单位电流系数	BG_I_LC<3:0>	单位电流系数
0	(32/32)I	0	(32/32)I
1	(33/32)I	1	(31/32)I
2	(34/32)I	2	(30/32)I
3	(35/32)I	3	(29/32)I
4	(36/32)I	4	(28/32)I
5	(37/32)I	5	(27/32)I
6	(38/32)I	6	(26/32)I
7	(39/32)I	7	(25/32)I
8	(40/32)I	8	(24/32)I
9	(41/32)I	9	(23/32)I
10	(42/32)I	10	(22/32)I
11	(43/32)I	11	(21/32)I
12	(44/32)I	12	(20/32)I
13	(45/32)I	13	(19/32)I
14	(46/32)I	14	(18/32)I
15	(47/32)I	15	(17/32)I

注：当 BG_I_LC = 0, BG_I_MC = 15 时，ENS001 内置单元电流可设置为最大，最高可输出 120mA 刺激电流。

5 系统控制器（SCU）

5.1 PMU

默认情况下，系统或上电复位后，MCU 处于运行模式。在运行模式下，CPU 由 HCLK 计时(默认是 HSI RC 的 8MHz)，并执行程序代码。当 CPU 不需要保持运行时，有几种低功耗模式可用于节省电力。当某些应用不使用 RTC/LCD 时，可以通过软件禁用 LSI RC(约 32kHz)和 LSE OSC(带外部晶振，精确 32.768kHz)，以节省电力。在 SYSCTRL 寄存器中也有外设启用位，如果一些外设没有使用，用户可以禁用外设来门控他们的时钟。

支持五种电源模式：

- 运行模式：正常运行模式，CPU 以高频时钟运行，所有外设均可激活。
- 睡眠模式：HCLK 被门控，CPU/所有 AHB 外设时钟关闭，所有 APB 外设和核心外设(如 NVIC、SysTick 等)都可以在中断或事件发生时运行并唤醒 CPU。
- 低功耗运行模式：系统时钟(SYSCLK)切换到大约 32KHz 的 LFCLK，为了节省电力，用户可以将程序复制到 SRAM 中执行。
- 低功耗睡眠模式：系统时钟(SYSCLK)切换到大约 32KHz 的 LFCLK，和睡眠模式一样，HCLK 是门控的。
- 停止模式：禁用 HSI RC，LFCLK 可以一直运行。
- 此外，可以通过以下方法之一来降低运行模式下的功耗：
  - 降低系统时钟。
  - 当未使用 APB 和 AHB 外设时，门控时钟。

如果 SCR 的 SLEEPONEXIT 位设置为 1，当处理器完成异常处理程序的执行并返回到线程模式时，它立即进入睡眠模式。在只在发生中断时需要处理器运行的应用程序中使用此机制。

通常，处理器只有在检测到具有足够优先级的异常导致异常输入时才会被唤醒。

一些嵌入式系统可能必须在处理器唤醒之后，在执行中断处理程序之前执行系统恢复任务。要实现这一点，请将 PRIMASK 位设置为 1。如果到达已启用的中断，且其优先级高于当前异常优先级，则处理器会醒来，但不会执行中断处理程序，直到处理器将 PRIMASK 初始屏蔽设置为零。

表 7 低功耗模式汇总

模式名称	条目	唤醒源	唤醒系统时钟	对时钟的影响
睡眠（现在睡眠或退出睡眠）	WFI 或从 ISR 返回	任何中断	与进入睡眠模式前相同	HCLK 关
	WFE	唤醒事件		
低功耗运行	SYSCLK_SEL 设置为 2’b1x	SYSCLK_SEL 设置为 2’b0x	无变化	无
低功耗睡眠	SYSCLK_SEL 设置为 2’b1x + WFI 或从 ISR 返回	任何中断	与进入低功耗睡眠模式前相同	HCLK 关
	SYSCLK_SEL 设置为 2’b1x + WFE	唤醒事件		
停止	SLEEPDEEP 位+ WFI 或从 ISR 或 WFE 返回	任何 EXTI 行(在 EXTI 寄存器中配置)特定的外设事件	HSI RC	除 LSI 和 LSE 外，所有时钟关闭

表 8 取决于工作模式的功能

功能	运行	睡眠	低功耗运行	低功耗睡眠	停止
CPU	Y	-	Y	-	-
MTP 内存	Y	-	Y	-	-
SRAM	Y	-	Y	-	-
HSI	O	O	O	O	-
LSI	O	O	O	O	O
LSE	O	O	O	O	O
LVD	O	O	O	O	O
COMP0/1	O	O	O	O	O
ADC	O	O	O	O	-
波形信号发生器	O	O	O	O	-
RTC	O	O	O	O	O
LCD	O	O	O	O	O
UART0/1	O	O	O	O	-
I2C0/1	O	O	O	O	-
SPI0/1	O	O	O	O	-
定时器/ 双定时	O	O	O	O	-
WDT	O	O	O	O	-
PWM	O	O	O	O	-
GPIOs	O	O	O	O	O

Y = 确定 O = 可选

5.2 复位控制

ENS1 复位控制包括电源复位和系统复位两种复位控制。上电复位，称为冷复位，在上电期间重置整个系统。系统复位重置处理器除 DBG 控制和 RTC 以外的核心和外设 IP 组件。

电源复位由电源接通复位和电源关闭复位(POR/PDR 复位）产生。电源复位将所有寄存器设置为其复位值。当内部 LDO 稳压器准备提供 1.8V 电源时，有功信号较低的功率复位将被取消。下电时，PDR 阈值电压可配置为 xxx(待定例)。RESET 复位服务例程向量固定在内存映射中的地址 0x0000_0004。

调试逻辑和 RTC 只能通过 POR/PDR 复位。

系统复位由以下事件引起：

- 上电复位
- 外部复位引脚
- 看门狗定时器复位
- Cortex-M0 AIRCR 寄存器中的 SYSRESETREQ 位被设置为软复位。
- CPU 锁定(当 CPU 处于 HardFault 或 NMI 处理程序中时，会发生另一个 HardFault 事件)

5.3 时钟控制

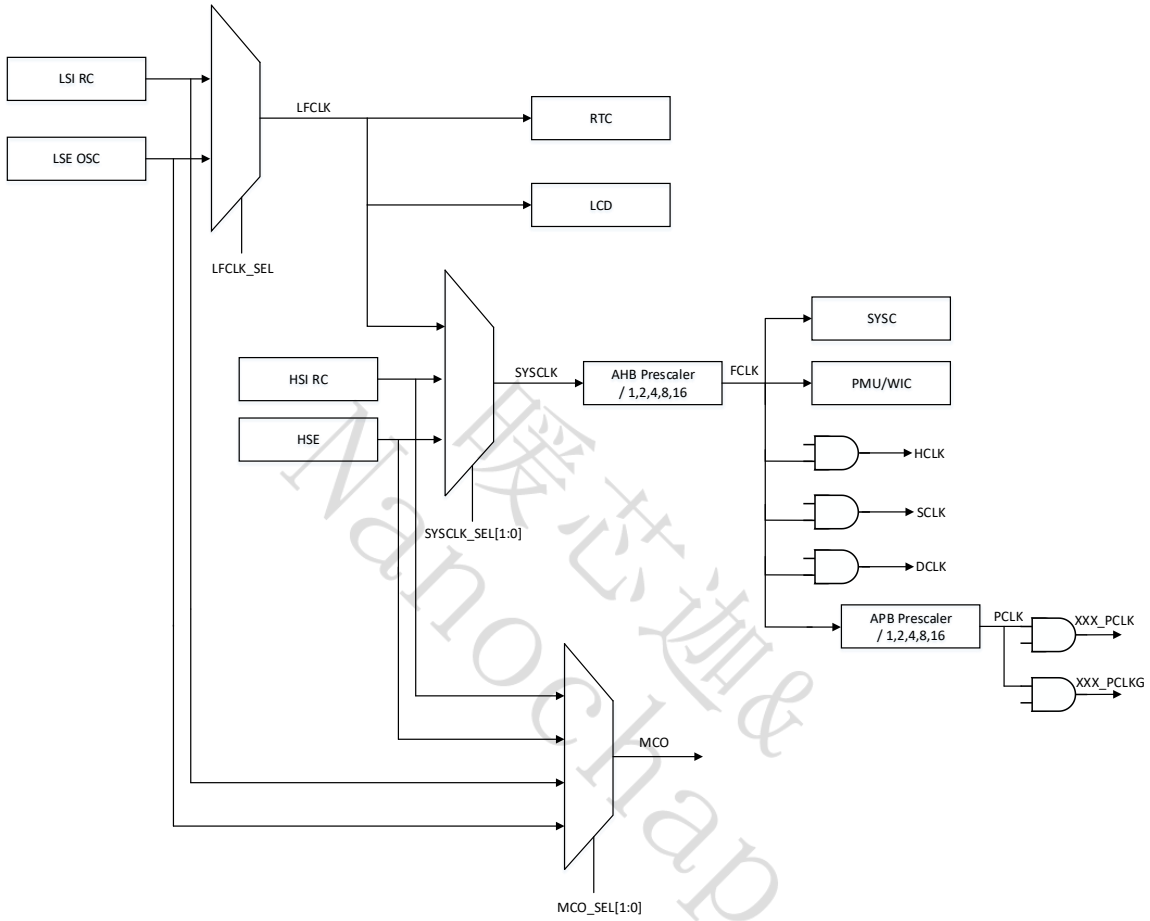
5.3.1 时钟树结构

可以使用四种不同的时钟源：

- HSI RC -高速全集成 RC 振荡器产生 HSI 时钟(约 4-32MHz)
- HSE - 高速外部时钟(高达 32MHz)
- LSI RC - 低速全集成 RC 振荡器产生 LSI 时钟(约 32KHz)
- LSE OSC - 带外部晶体谐振器的低速振荡器(精确 32.768KHz)

时钟树结构如下：

图 7 时钟树



可以选择以下时钟之一作为系统时钟(SYSCLK)：

- LFCLK (来自 LSI 或 LSE)
- HSI
- HSE

系统时钟最大频率为 32MHz。系统复位时，将选择 HSI 时钟作为系统时钟。

只有当目标时钟源准备好时，才会发生时钟源切换。

LFCLK 可以从 LSI 或 LSE 中选择。可以通过软件禁用 LSI 和 LSE 时钟。

对于 LSE 振荡器，正常运行时，将 LSE_OSCEN 和 LSE_CLKEN 设置为 1。在待机状态下，将 LSE_CLKEN 设置为 0。可以禁用 XTALOUTCORE 的时钟输出，同时保持振荡器核心运行。如果需要通过输入引脚将时钟信号驱动到芯片中进行测试，则在单元中包含一个旁路多路复用器，将 INVINPAD 驱动的时钟信号直接路由到 XTALOUTCORE。要启用此功能，请将 OSCEN 设置为 0，这也将关闭振荡器核心。

5.3.2 不同电源模式下的时钟说明

AHB 和 APB 外设时钟可以通过软件禁用。

睡眠和低功率睡眠模式停止 HCLK。

停止模式停止所有时钟，除 LSI 和 LSE 外，HSI RC 被禁用，当离开停止模式时，HSI 自动成为系统时钟。

MCO 引脚输出，彼此独立，从 LSI/LSE/HSI/HSE 中选择时钟。

下表显示了每个低功耗模式下的时钟状态(开/关/用户)。

表 9 不同电源模式下的时钟描述

时钟	运行	LP 运行	睡眠	LP 睡眠	停止	描述
LSI	用户	开	用户	用户	用户	32KHz 内 RC 振荡器
LSE		选择取决于用户				32.768KHz 晶体振荡器
LFCLK						从 LSI 和 LSE 切换低频时钟
HSI	开	用户	开	开	关	HSI RC 4-32MHz
HSE	选择取决于用户	用户	选择取决于用户	选择取决于用户	用户	高达 32MHz 的外部时钟
SYSCLK	开	开	开	开	关	从 HSI，HSE 和 LFCLK 切换系统时钟
FCLK	开	开	开	开	关	从 SYSCLK 分离的 AHB 自由运行的时钟
HCLK	开	开	关	关	关	用于 CPU/MTP/SRAM/GPIO 的 AHB 总线时钟
SCLK	开	开	开	开	关	用于 NVIC 的 CPU 核心系统时钟
DCLK	动态	动态	动态	动态	关	CPU 核心调试时钟，当没有调试请求时，它被门控
PCLK	开	开	开	开	关	APB 自由运行时钟
PCLKG	动态	动态	关	关	关	APB 门控总线时钟，当没有 APB 周期时，它是门控的

5.3.3 时钟开关顺序

当 SYSCLK 从 HSI 切换到 HSE 时，请按如下顺序配置

1) 确保 HSE 时钟正常运行
2) 将 SYSCLK_SEL 设置为 2'b01
3) 轮询 SYSCLK_SWSTS 为 2'b01
4) 将 HSI_EN 设置为 0

当将 SYSCLK 从 HSI 切换到 LFCLK 时，按以下顺序配置

1) 确保 LFCLK 运行正常，如需更换 LFCLK 源，请先执行 LFCLK 开关顺序
2) 将 SYSCLK_SEL 设置为 2’b1x
3) 轮询 SYSCLK_SWSTS 为 2’b1x
4) 将 HSI_EN 设置为 0

当 LFCLK 从 LSI 切换到 LSE 时，请按照如下顺序进行配置

1) 将 LSE_OSCEN 和 LSE_CLKEN 均设置为 1
2) 等待 LSE 稳定(稳定时间待定)
3) 将 LFCLK_SEL 设置为 1
4) 轮询 LFCLK_SWSTS 为 1
5) 将 LSI_EN 设置为 0

5.4 寄存器

表 10 SYSCTRL 寄存器

Offset	首字母缩写	寄存器描述
00h	CLK_CFG	时钟配置寄存器
04h	HSI_CTRL	HSI 控制寄存器
08h	LSI_CTRL	LSI 控制寄存器
0Ch	LSE_CTRL	LSE 控制寄存器
10h	AHB_CLKEN	AHB 外设时钟启用寄存器
14h	APB_CLKEN	APB 外设时钟启用寄存器
18h	PERI_CLKEN	外设工作时钟启用寄存器
1Ch	SLP_PCLKEN	在睡眠/停止模式下启用 APB 外设时钟寄存器
20h	RST_CTRL	复位控制寄存器
24h	RST_FLAG	复位标志寄存器
28h	PRST_KEY	外设复位启用密钥寄存器
2Ch	AHB_RST	AHB 外设复位寄存器
30h	APB_RST	APB 外设复位寄存器
34h	SYS_CFG	系统配置寄存器
38h	PMU_CTRL	PMU 控制寄存器

偏移地址： 03-00h

时钟配置寄存器

位	字段名	属性	默认	字段描述
31：18	-	RO	0	预留的
17：16	MCO_SEL	RW	0	MCU 时钟输出选择 00： HSI 01： HSE 10： LSI 11： LSE
15	-	RO	0	预留的

14：12	APB_PRES	RW	0	APB 时钟预分频 0xx： PCLK 未分频 100： PCLK 2 分频 101： PCLK 4 分频 110： PCLK 8 分频 111： PCLK 16 分频
11	-	RO	0	预留的
10：8	AHB_PRES	RW	0	AHB 时钟预分频 0xx： HCLK 未分频 100： HCLK2 分频 101： HCLK4 分频 110： HCLK 8 分频 111： HCLK 16 分频
7：6	-	RO	0	预留的
5	LFCLK_SWSTS	RO	0	LFCLK 开关状态 0： LSI 为 LFCLK 1： LSE 为 LFCLK
4	LFCLK_SEL	RW	0	LFCLK 开关选择 0： LSI 为 LFCLK 1： LSE 为 LFCLK
3：2	SYSCLK_SWSTS	RO	0	系统时钟开关状态 00： RCHF 为系统时钟 01： EXTHF 为系统时钟 1x： LFCLK 为系统时钟

1：0	SYSCLK_SEL	RW	0	系统时钟开关选择 00： HSI 为系统时钟 01： HSE 为系统时钟 1x： LFCLK 为系统时钟 当 MCU 退出停止模式时，硬件强制设置为 00 (HSI 选择)。

偏移地址： 07-04h

HSI 控制寄存器

位	字段名	属性	默认	字段描述
31：6	-	RO	0	预留的
5：4	HSI_FREC	RW	01b	HSI OSC 频率选择 00： 4MHz 01： 8MHz 10： 16MHz 11： 32MHz
3：1	-	RO	0	预留的
0	HSI_EN	RW	1	HSI OSC 时钟启用 由软件设置和清除 当进入停止模式时，通过硬件清除以停止 HSI OSC 当 HSI OSC 直接或间接用作系统时钟时(当离开停止模式时)，硬件强制保持 HSI OSC ON 0： HSI OSC OFF 1： HSI OSC ON

偏移地址： 0B-08h

LSI 控制寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	LSI_EN	RW	1	LSI 启用位 0： 禁用 1： 启用

偏移地址： 0F-0Ch

LSE 控制寄存器

位	字段名	属性	默认	字段描述
31：29	-	RO	0	预留的
1	LSE_CLKEN	RW	0	LSE 时钟启用位 0： 禁用时钟输出 1： 启用时钟输出
0	LSE_OSCEN	RW	0	LSE 振荡器启用位 0： 关闭振荡器核心 1：开启振荡器核心

偏移地址： 13-10h

AHB 外设时钟启用寄存器

位	字段名	属性	默认	字段描述
31：4	-	RO	0	预留的
2：0	AHB_PORT_CLKEN	RW	111b	位 2： MTP 寄存器访问 HCLK 启用位 位 1： EXTI HCLK 启用位 位 0： GPIO HCLK 启用位

偏移地址： 17-14h

APB 外设时钟启用寄存器

位	字段名	属性	默认	字段描述
31：16	-	RO	0	预留的
15：0	APB_PORT_CLKEN	RW	0	位 15： RTC PCLK 启用位 位 14： 模拟控制 PCLK 启用位 位 13： ADC 控制 PCLK 启用位 位 12： 波形发生器 PCLK 启用位 位 11： LCD 驱动 PCLK 启用位 位 10： 双定时器 PCLK 启用位 位 9： 定时器 1 PCLK 启用位 位 8： 定时器 0 PCLK 启用位 位 7： PWM PCLK 启用位 位 6： WDT PCLK 启用位 位 5： I2C1 PCLK 启用位 位 4： I2C0 PCLK 启用位 位 3： SPI1 PCLK 启用位 位 2： SPI0 PCLK 启用位 位 1： UART1 PCLK 启用位 位 0： UART0 PCLK 启用位

偏移地址： 1B-18h

外设工作时钟启用寄存器

位	字段名	属性	默认	字段描述
31：2	-	RO	0	预留的
1	LCD_CLKEN	RW	0	LCD 工作时钟启用 0： 禁用 1： 启用

0	RTC_CLKEN	RW	0	RTC 工作时钟启用 0： 禁用 1： 启用

偏移地址： 1F-1Ch

APB 在睡眠/停止模式下启用外设时钟寄存器

位	字段名	属性	默认	字段描述
31：16	-	RO	0	预留的
15：0	APB_SLP_CLKEN	RW	0xFFFF	位 15： RTC PCLK 在睡眠和停止模式下启用 位 14： 模拟控制 PCLK 在睡眠和停止模式下启用 位 13： ADC 控制 PCLK 在睡眠和停止模式下启用 位 12： 波形发生器 PCLK 在睡眠和停止模式启用 位 11：LCD 驱动器 PCLK 在睡眠和停止模式下启用 位 10： 双定时器 PCLK 在睡眠和停止模式下启用 位 9： 定时器 1 PCLK 在睡眠和停止模式下启用 位 8： 定时器 0 PCLK 在睡眠和停止模式下启用 位 7： PWM PCLK 在睡眠和停止模式下启用 位 6： WDT PCLK 在睡眠和停止模式下启用 位 5： I2C1 PCLK 在睡眠和停止模式下启用 位 4： I2C0 PCLK 在睡眠和停止模式下启用 位 3： SPI1 PCLK 在睡眠和停止模式下启用 位 2： SPI0 PCLK 在睡眠和停止模式下启用 位 1： UART1 PCLK 在睡眠和停止模式下启用 位 0： UART0 PCLK 在睡眠和停止模式下启用

偏移地址： 23-20h

复位控制寄存器

位	字段名	属性	默认	字段描述
31：30	-	RO	0	预留的
0	LOCKUP_RESETEN	RW	0	CPU 锁定复位启用位 0： 禁用 1： 启用

偏移地址： 27-24h

复位标志寄存器

位	字段名	属性	默认	字段描述
31：5	-	RO	0	预留的
4	PINRST_FLAG	RW1C	0	复位引脚标志 该位由 NRST 引脚复位来设置 通过写入此位或通过 POR 来清除 0： 未发生复位引脚低电平 1： 发生复位引脚低电平
3	LOCKUPRST_FLAG	RW1C	0	锁定复位标志 当锁定复位发生时，该位由硬件设置 在这个位上写入 1，或者通过 POR 来清除 0： 未发生锁定复位 1： 发生锁定复位
2	WDTRST_FLAG	RW1C	0	看门狗复位标志 当看门狗复位发生时，该位由硬件设置 在这个位上写入 1，或者通过 POR 来清除 0： 未发生看门狗复位 1： 发生看门狗复位

1	SOFTRST_FLAG	RW1C	0	软件复位标志 当软件在 NVIC 空间中写入 SYSRESETREQ 时，该位由硬件设置 在这个位上写入 1，或者通过 POR 来清除 0： 未发生软件复位 1： 发生软件复位
0	PORRST_FLAG	RW1C	1	POR/PDR 复位标志 当 POR/PDR 复位发生时，该位由硬件设置 在这位上写 1 就可以清除 0： 未发生 POR/PDR 复位 1： 发生 POR/PDR 复位

偏移地址： 2B-28h

外设复位启用密钥寄存器

位	字段名	属性	默认	字段描述
31：0	PRST_KEY	WO	0	外设复位启用密钥寄存器 写入“0x1A2B_3C4D”表示开启外设软复位功能，写入其他值表示禁用外设软复位功能 只写，读取此寄存器返回 0

偏移地址： 2F-2Ch

AHB 外设复位寄存器

位	字段名	属性	默认	字段描述
1	-	RO	0	预留的
2：0	AHB_PORT_RST	RW	0	高电平 位 2： MTP 寄存器访问复位位 位 1： EXTI 复位位 位 0： GPIO 复位位

偏移地址： 33-30h

APB 外设复位寄存器

位	字段名	属性	默认	字段描述
31：16	-	RO	0	预留的
15：0	APB_PORT_RST	RW	0	高电平 位 15： RTC 复位位 位 14： 模拟控制复位位 位 13： ADC 控制复位位 位 12： 波形发生器复位位 位 11： LCD 驱动复位位 位 10： 双定时器复位位 位 9： 定时器 0 复位位 位 8： 定时器 1 复位位 位 7： PWM 复位位 位 6： WDT 复位位 位 5： I2C1 复位位 位 4： I2C0 复位位 位 3： SPI1 复位位 位 2： SPI0 复位位 位 1： UART1 复位位 位 0： UART0 复位位

偏移地址： 37-34h

系统配置寄存器

位	字段名	属性	默认	字段描述
31：10	-	RO	0	预留的

9：8	BOOT_MODE	RO	0	引导引脚状态位选择的引导模式 该位为只读。表明 BOOT[1：0]引脚选的引导模式 00： 基地址引导模式的 MTP 主阵列 01： MTP 高 4K 字节引导模式 1x： 嵌入式 SRAM 引导模式
7：2	-	RO	0	预留的
1：0	REMAP	RW	0	内存映射选择位 这些位是由软件设置和清除的。这个位控制地址 0x0000_0000 的内部映射。复位后，这些位在 BOOT 引脚选择的内存映射上 00： MTP 主阵列的基址在 0x0000_0000 的映射 01： MEP 高 4K 字节在 0x0000_0000 的映射 1x： SRAM 在 0x0000_0000 的映射

偏移地址： 3B-38h

PMU 控制寄存器

位	字段名	属性	默认	字段描述
31：6	-	RO	0	预留的
5：4	WKUP_DLY	RW	0	从停止模式唤醒时延迟时间 00： 1us 01： 2us 10： 4us 11： 8us
3：1	-	RO	0	预留的
0	WKFREQ_SEL	RW	0	从停止模式唤醒时选择 HSI 频率 0： 与进入停止模式前相同 1： 8MHz

6 CPU

6.1 概述

Cortex-M0 处理器是一个入门级的 32 位 ARM Cortex 处理器，专为广泛的嵌入式应用程序而设计。 它建立在一个高度区域和功率优化的 32 位处理器核心上，具有 3 级管道冯·诺伊曼架构。Cortex-M0 处理器实现了 ARMv6-M 架构，该架构基于 16 位 Thumb 指令集，并包含具有高代码密度的 Thumb-2 技术。

6.2 处理器配置

表 11 处理器配置

功能	选择项	配置
中断	1~32	32
数据字节序	小/大	小
SysTick 定时器	存在或不存在	存在
监视点	0~2	2
断点	0~4	4
调试	存在或不存在	存在
JTAGnSW	JTAG 或 SWD	SWD
乘法器	快或小	快(单循环)
复位所有寄存器	是或否	是
WIC	存在或不存在	存在

6.3 核心寄存器

表 12 核心寄存器

名称	类型	描述
R0-R12	RW	通用寄存器
SP (R13)	RW	堆栈指针，SP 可以通过配置控制寄存器选择为 MSP 或 PSP 处理模式： MSP (主堆栈指针)

		线程模式： PSP (进程堆栈指针)
LR (R14)	RW	链接寄存器，它存储子例程、函数调用和异常的返回信息
PC (R15)	RW	程序计数器，它包含当前程序地址
APSR	RW	应用程序状态寄存器，它包含条件标志的当前状态
IESR	RO	中断程序状态寄存器，它包含当前 ISR 的异常号
EPSR	RO	执行程序状态寄存器，它包含 Thumb 状态位
PRIMASK	RW	优先级屏蔽寄存器，它可以防止激活所有具有可配置优先级的异常
CONTROL	RW	控制处理器处于线程模式时使用的堆栈

6.4 异常和中断

系统异常和中断由 NVIC 管理。中断管理功能由位于 SCS 连接的 PPB 总线内的若干可编程寄存器控制。NVIC：

- 支持 32 个中断和 1 个 NMI
- 灵活的中断管理(末尾连锁和迟到异常)
- 处理嵌套中断
- 矢量异常入口
- 中断屏蔽
- 四种中断优先级

当接受异常时，R0-R3、R12、R14、PC 和 xPSR 被自动推送到当前堆栈内存，LR 被更新为 EXC_RETURN，在异常返回时使用。

在异常处理过程结束时，如果没有任何其他异常需要处理，则恢复先前存储在堆栈内存中的寄存器值，并恢复中断的程序。

如果 CPU 处于 HardFault 或 NMI 处理过程中，再次发生 HardFault，CPU 将进入锁定状态，CPU 核心将被重置，如果设置了 LOCKUPRESET，则允许系统复位。

表 13 中断向量表

位置	IRQ 号	异常类型	优先级	向量地址
0	-	初始 SP 值	-	0x0000_0000
1	-	复位	-3，最高级	0x0000_0004
2	-14	NMI (WDT)	-2	0x0000_0008
3	-13	Hardfault	-1	0x0000_000C
4-10	-	预留的	-	-
11	-5	SVC	可配置	0x0000_002C
12-13	-	预留的	-	-
14	-2	PendSV	可配置	0x0000_0038
15	-1	SysTick	可配置	0x0000_003C
16	0	LVD (EXTI line 24)	可配置	0x0000_0040
17	1	RTC	可配置	0x0000_0044
18	2	COMP0 (EXTI line 25)	可配置	0x0000_0048
19	3	COMP1 (EXTI line 26)	可配置	0x0000_004C
20	4	GPIO0_7 (EXTI line 0-7)	可配置	0x0000_0050
21	5	GPIO8_15 (EXTI line 8-15)	可配置	0x0000_0054
22	6	GPIO16_23 (EXTI line 16-23)	可配置	0x0000_0058
23	7	MTP	可配置	0x0000_005C
24	8	Charger_ok(EXTI line 27)	可配置	0x0000_0060
25	9	Charger_end(EXTI line 28)	可配置	0x0000_0064
26	10	ADC	可配置	0x0000_0068
27	11	LCD	可配置	0x0000_006C
28	12	UART0	可配置	0x0000_0070
29	13	UART1	可配置	0x0000_0074
30	14	SPI0	可配置	0x0000_0078
31	15	SPI1	可配置	0x0000_007C

32	16	I2C0 事件	可配置	0x0000_0080
33	17	I2C0 错误	可配置	0x0000_0084
34	18	I2C1 事件	可配置	0x0000_0088
35	19	I2C1 错误	可配置	0x0000_008C
36	20	PWM	可配置	0x0000_0090
37	21	基础 TIM0	可配置	0x0000_0094
38	22	基础 TIM1	可配置	0x0000_0098
39	23	双 TIM	可配置	0x0000_009C
40	24	过温(EXTI 29)	可配置	0x0000_00A0
41	25	-	可配置	0x0000_00A4
42	26	-	可配置	0x0000_00A8
43	27	-	可配置	0x0000_00AC
44	28	-	可配置	0x0000_00B0
45	29	-	可配置	0x0000_00B4
46	30	-	可配置	0x0000_00B8
47	31	-	可配置	0x0000_00BC

6.5 调试

基本调试功能包括：

- 处理器暂停，单步操作
- 处理器核心寄存器访问
- 四个硬件断点
- 两个监视点
- 无限软件断点(BKPT 指令)
- 全系统内存访问
- SWD 接口

7 Cortex-M0 外设

7.1 概述

私有外设总线(PPB)的地址映射为：

表 14 核心外设寄存器区

地址	核心外设
0xE000E008-0xE000E00F 0xE000ED00-0xE000ED3F	系统控制块
0xE000E010-0xE000E01F	SysTick 定时器
0xE000E100-0xE000E4EF 0xE000EF00-0xE000EF03	嵌套矢量中断控制器

7.2 嵌套矢量中断控制器

本节描述了嵌套矢量中断控制器(NVIC)和它使用的寄存器。NVIC 支持：

- 1 ~ 32 个中断。
- 每个中断的可编程优先级为 0-192，步骤为 64。较高的级别对应较低的优先级，因此级别 0 是最高的中断优先级。
- 中断信号的电平和脉冲检测。
- 中断末尾连锁。
- 外部不可屏蔽中断(NMI)。

处理器在异常条目上自动堆叠其状态，并在异常退出时解除堆栈此状态，没有指令执行时间。这提供了低延迟的异常处理。

7.2.1 寄存器

表 15 NVIC 寄存器

地址	首字母缩写	类型	复位值	寄存器描述
0xE000E100	ISER	RW	0x00000000	中断设置启用寄存器
0xE000E180	ICER	RW	0x00000000	中断清除启用寄存器
0xE000E200	ISPR	RW	0x00000000	中断设置挂起存器
0xE000E280	ICPR	RW	0x00000000	中断解挂寄存器
0xE000E400-0xE000E41C	IPR0-7	RW	0x00000000	中断优先寄存器

地址： 0xE000E100

中断启用设置寄存器

位	字段名	属性	默认	字段描述
31：0	SETENA	RW	0	中断设置启用位 写入： 0 = 无作用 1 = 启用中断 读取： 0 = 中断禁用 1 = 中断启用

地址： 0xE000E180

中断启用设置寄存器

位	字段名	属性	默认	字段描述
31：0	CLRENA	RW	0	中断清除启用位 写入： 0 = 无作用 1 = 禁用中断 读取： 0 = 中断禁用 1 =中断启用

地址： 0xE000E200

中断解挂寄存器

位	字段名	属性	默认	字段描述
31：0	CLRPEND	RW	0	中断解挂位 写入： 0 = 无作用 1 = 删除挂起状态的中断 读取： 0 =中断未挂起 1 = 中断挂起 向 ICPR 位写入 1 不会影响相应中断的活动状态

地址： 0xE000E280

中断设置挂起寄存器

位	字段名	属性	默认	字段描述
31：0	SETPEND	RW	0	中断挂起控制位 写入： 0 = 无作用 1 = 将中断状态更改为挂起 读取： 0 = 中断未挂起 1 =中断挂起 将 1 写入与对应的 ISPR 位： ■ 挂起中断无作用 ■ 一个被禁用的中断将中断的状态设置为挂起

地址： 0xE000E400-0xE000E41C

中断优先级寄存器

位	字段名	属性	默认	字段描述
31：24	PRI_(4n+3)	RW	0	每个优先级字段都有一个优先级值，0-192。该值越低，对应中断的优先级越高。处理器只实现了每个字段的位[7：6]，位[5：0]读为零，写忽略。 这意味着将 255 写入优先级寄存器将值保存到 192。
23：16	PRI_(4n+2)	RW	0	
15：8	PRI_(4n+1)	RW	0	
7：0	PRI_(4n)	RW	0	

7.3 系统控制块

SCB (系统控制块)提供系统实现信息和系统控制。这包括系统异常的配置、控制和报告。

表 16 SCB 寄存器

地址	首字母缩写	类型	复位值	寄存器描述
0xE000ED00	CPUID	RO	0x410CC200	CPUID 寄存器
0xE000ED04	ICSR	RW	0x00000000	中断控制和状态寄存器
0xE000ED0C	AIRSR	RW	0xFA050000	应用中断和复位控制寄存器
0xE000ED10	SCR	RW	0x00000000	系统控制寄存器
0xE000ED14	CCR	RO	0x00000204	配置和控制寄存器
0xE000ED1C	SHPR2	RW	0x00000000	系统处理优先级寄存器 2
0xE000ED20	SHPR3	RW	0x00000000	系统处理优先级寄存器 3

地址： 0xE000ED00

CPUID 寄存器

位	字段名	属性	默认	字段描述
31：24	实现器	RO	0x41	实现器代码： 0x41 = ARM
23：20	变异数	RO	0	变异数： 0x0 = 修订 0
19：16	常数	RO	0xC	定义处理器体系结构的常数： 0xC = ARMv6-M 架构
15：4	部件号	RO	0xC20	处理器的部件号： 0xC20 = Cortex-M0

3：0	修订	RO	0	修订号： 0x0 =补丁 0

地址： 0xE000ED04

中断控制与状态寄存器

位	字段名	属性	默认	字段描述
31	NMIPENDSET	RW	0	NMI 设置挂起位 写入： 0 = 无作用 1 = 将 NMI 异常状态变更为挂起 读取： 0 = NMI 异常不挂起 1 = NMI 异常挂起 因为 NMI 是最高优先级的异常，通常处理器一旦检测到对这个位写 1，就会进入 NMI 异常处理程序。然后进入处理程序将此位清除为 0。这意味着只有当处理器 ix 执行 NMI 异常处理程序时，NMI 信号被重新声明时，NMI 异常处理程序对该位的读取才返回 1。
30：29	-	RO	0	预留的
28	PENDSVSET	RW	0	PendSV 设置挂起位 写入： 0 = 无作用 1 = 将 PendSV 异常状态变更为挂起 读取： 0 = PendSV 异常不挂起 1 = PendSV 异常挂起 在这个位上写 1 是设置 PendSV 异常状态变更为挂起的唯一方式。

27	PENDSVCLR	WO	0	PendSV 解挂位 写入： 0 = 无作用 1 = 从 PendSV 异常中移除挂起状态
26	PENDSTSET	RW	0	SysTick 异常设置挂起位 写入： 0 = 无作用 1 = 将 sysTick 异常状态变更为挂起 读取： 0 = SysTick 异常不挂起 1 = SysTick 异常挂起
25	PENDSTCLR	WO	0	SysTick 异常解挂位 写入： 0 = 无作用 1 =从 SysTick 异常中移除挂起状态
24：23	-	RO	0	预留的
22	ISRPENDING	RO	0	中断挂起标志，不包括 NMI 和故障： 0 =中断未挂起 1 = 中断挂起
21：18	-	RO	0	预留的
17：12	VECTPENDING	RO	0	最高优先级挂起的启用异常的异常号： 0 = 没有挂起的异常 非零=最高优先级挂起的启用异常的异常号。
11：6	-	RO	0	预留的
5：0	VECTACTIVE	RO	0	包含活动异常号： 0 = 线程模式 非零=当前活动异常的异常号

地址： 0xE000ED0C

应用中断和复位控制寄存器

位	字段名	属性	默认	字段描述
31：16	VECTKEY	RW	0xFA05	寄存器密钥 写入时，将 0x05FA 写入 VECTKEY，否则写入将被忽略 读取为 0xFA05
15	ENDIANESS	RO	0	实现了数据字节顺序： 0 = 小端 1 = 大端
14：3	-	RO	0	预留的
2	SYSRESETREQ	WO	0	系统复位请求： 0 = 无作用 1 =请求系统级复位 该位读取为 0
1	VECTCLRACTIVE	WO	0	仅供调试使用，这个位读为 0。当写入寄存器时，你必须将 0 写入这个位，否则行为是不可预测的。
0	-	RO	0	预留的

地址： 0xE000ED10

系统控制寄存器

位	字段名	属性	默认	字段描述
31：5	-	RO	0	预留的

4	SEVONPEND	RW	0	发送事件挂起位： 0 = 只有启用的中断或事件才能唤醒处理器，禁用的中断将被排除在外 1 = 启用的事件和所有中断(包括禁用的中断)都可以唤醒处理器 当事件或中断进入挂起状态时，事件信号将从 WFE 唤醒处理器。如果处理器没有等待事件，则该事件将被注册并影响下一个 WFE。 处理器也会在 SEV 指令或外部事件执行时被唤醒。
3	-	RO	0	预留的
2	SLEEPDEEP	RW	0	控制处理器是否使用睡眠或深度睡眠作为低功耗模式： 0 =睡眠 1 = 深度睡眠
1	SLEEPONEXIT	RW	0	处理模式返回线程模式时表明退出后睡眠： 0 =返回线程模式时不要睡眠 1 = 从 ISR 返回线程模式进入睡眠或深睡眠 将该位设置为 1 可以使中断驱动的应用程序避免返回空的主应用程序
0	-	RO	0	预留的

地址： 0xE000ED14

配置和控制寄存器

位	字段名	属性	默认	字段描述
31：10	-	RO	0	预留的
9	STKALIGN	RO	1	总是读取为 1，指示异常条目上的 8 字节堆栈对齐。在异常进入时，处理器使用堆叠 PSR 的[9]位来指示堆栈对齐。在从异常返回时，它使用这个堆叠位来恢复正确的堆栈对齐。
8：4	-	RO	0	预留的
3	UNALIGN_TRP	RO	1	始终读取为 1，表示所有未对齐的访问都会生成硬故障。
2：0	-	RO	0	预留的

地址： 0xE000ED1C

系统处理优先级寄存器 2

位	字段名	属性	默认	字段描述
31：24	PRI_11	RW	0	系统处理程序 11 SVCALL 的优先级
23：0	-	RO	0	预留的

地址： 0xE000ED20

系统处理优先级寄存器 3

位	字段名	属性	默认	字段描述
31：24	PRI_15	RW	0	系统处理程序优先级 15,SysTick 异常
23：16	PRI_14	RW	0	系统处理程序 14 的优先级，PendSV
15：0	-	RO	0	预留的

7.4 SysTick 定时器

当启用时，定时器从重新加载值开始倒数到零，在下一个时钟周期中重新加载(包装到)SYST_RVR 中的值，然后在随后的时钟周期中递减。将 0 的值写入 SYST_RVR 将在下一次换行时禁用计数器。当计数器转换为 0 时，COUNTFLAG 状态位被设置为 1。读取 SYST_CSR 将 COUNTFLAG 位清除为 0。当处理器停止调试时，计数器不会递减。

软件可以配置 SysTick 定时器选择 SCLK 作为它的时钟源，或一个替代时钟源：FCLK 时钟 8 分频。

7.4.1 寄存器

表 17 SysTick 定时器寄存器

地址	首字母缩写	类型	复位值	寄存器描述
0xE000E010	SYST_CSR	RO	0x00000000	SysTick 控制与状态寄存器
0xE000E014	SYST_RVR	RW	0x00000000	SysTick 重新加载值寄存器
0xE000E018	SYST_CVR	RW	0x00000000	SysTick 当前值寄存器
0xE000E01C	SYST_CALIB	RW	0x00000000	SysTick 校准值寄存器

地址： 0xE000E010

SysTick 控制与状态寄存器

位	字段名	属性	默认	字段描述
31：17	-	RO	0	预留的
16	COUNTFLAG	RO	0	如果计时器自该寄存器最后一次读取以来计数为 0，则返回 1
15：3	-	RO	0	预留的
2	CLKSOURCE	RW	0	选择 SysTick 定时器源： 0 = 外部参考时钟 1 = 处理器时钟

1	TICKINT	RW	0	启用 SysTick 异常请求： 0 =倒数到零不会断言 SysTick 异常请求 1 = 倒数到零来断言 SysTick 异常请求
0	ENABLE	RW	0	启用计数器： 0 = 禁用计数器 1 = 启用计数器

地址： 0xE000E014

SysTick 控制与状态寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	RELOAD	RW	0	当计数器启用并且达到 0 时加载到 SYST_CVR 的值， RELOAD 值可以是 0x00000001-0x00FFFFFF 范围内的任何值。您可以编程值 0，但这无效，因为当从 1 计算到 0 时，SysTick 异常请求和计数标志被激活。 要生成具有 N 个处理器时钟周期的多次激发定时器，请使用重加载值 N-1。 例如，如果每 100 个时钟脉冲需要一次 SysTick 中断，则将 RELOAD 设置为 99。

地址： 0xE000E018

SysTick 当前值寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	CURRENT	RW	0	读取返回 SysTick 计数器的当前值 写入任何值都会清除字段 0，也会清除 SYST_CSR.COUNTFLAG 位到 0

地址： 0xE000E01C

SysTick 校准值寄存器

位	字段名	属性	默认	字段描述
31	NOREF	RO	0	读取为零，表示提供单独的参考时钟
30	SKEW	RO	1	读取为 1，10 ms 的校准值不准确
29：24	-	RO	0	预留的
23：0	TENMS	RO	0	校准值不可用

8 通用输入/输出 (GPIO)

8.1 概述

GPIO 块实现了一个具有 24 个 I/O 的 AHB-Lite 接口的通用 I/O 块，其中每个 I/O 都可以通过内存映射寄存器独立配置。通过 AHB 接口，所有寄存器都是可读可写的。

实际控制信号功能和计数可能会根据所选择的第三方 IOPAD 型号而发生变化。

8.1.1 功能列表

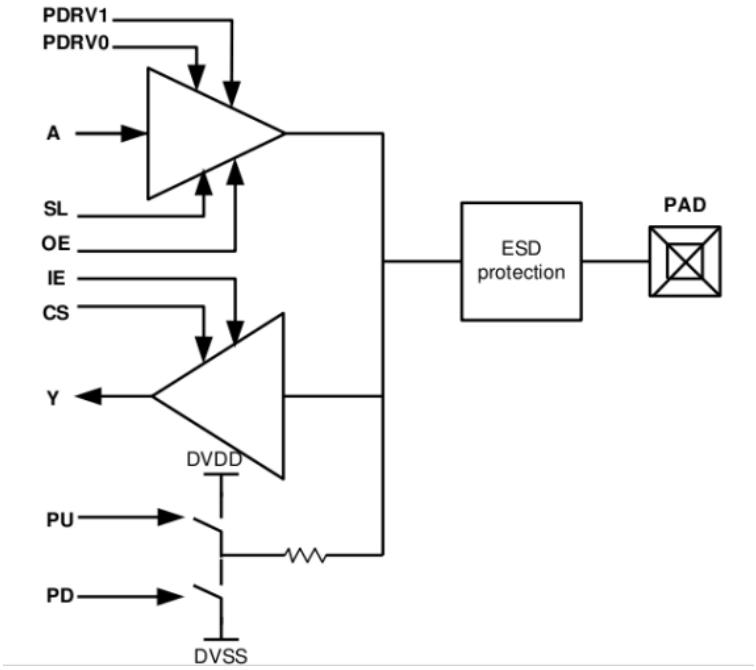
- 输入输出方向控制
- 每个引脚弱上拉/下拉
- 输出推挽/开漏启用控制
- 输出位设置/复位控件，支持原子读/修改写操作
- 输出驱动器强度选择
- 输入缓冲区选择(CMOS/施密特)
- 切换速率选择
- 模拟通道启用
- 替代函数
- 两个 HCLK 循环切换输出能力

8.2 函数描述

8.2.1 三态数字 I/O 单元

I/O 板是一个 5V 三态双向 I/O 板，具有 4mA，8mA，12mA 和 16mA 的可编程输出驱动强度。它可以被编程为 CMOS 输入或施密特触发器输入，带或不带上拉/下拉。在快速转换 16mA 驱动强度模式下，它可以在 100MHz 的频率下以 30pF 的电容性负载运行。控制引脚 PDRV0，PDRV1，SL，CS 的使用如寄存器表所示。

图 8 I/O 单元函数示意图



单元格的逻辑操作如下所示。

表 18 逻辑操作表

Driver Function

Input				Output
OE	PU	PD	A	PAD
0	0	0	X	Hi-Z
0	0	1	X	weak 0
0	1	0	X	weak 1
0	1	1	X	Hi-Z
1	X	X	0	0
1	X	X	1	1

Receiver Function

Input				Output
IE	PU	PD	PAD	Y
0	X	X	X	0
1	X	X	0	0
1	X	X	1	1
1	0	1	weak 0	0
1	1	0	weak 1	1

8.2.2 替代函数多路复用器

设备 I/O 引脚通过多路复用器连接到板上外设/模块，该多路复用器一次只允许一个外设备用功能连接到一个 I/O 引脚。这样，同一 I/O 引脚上可用的外设之间就不会发生冲突。

通过配置 GPIO_ALTF 寄存器，每个 I/O 有 4 个替代函数。GPIOx_ALTF[1：0]配置为 2'b11 时，pad 用于模拟函数。详细的替代函数映射信息显示在寄存器表中。

复位后多路复用器选择为替代函数 0。GPIO0/1 默认为调试引脚：SWCLK 和 SWDIO。复位后 SWCLK 为下拉，SWDIO 为上拉。

8.2.3 输入配置

当通过配置 GPIO_IE 寄存器将 GPIO 引脚配置为输入时：

- 可选择施密特触发器或 CMOS 输入
- 可以选择弱的上拉电阻或下拉电阻
- 在每个 AHB 时钟周期中，I/O 引脚上的数据被采样到输入数据寄存器中。
- 对输入数据寄存器的读访问提供了 I/O 状态

8.2.4 输出配置

当通过配置 GPIO_OE 寄存器将 GPIO 引脚配置为输出时：

- 可选择开漏模式拉拉模式

- 可以选择弱的上拉电阻或下拉电阻
- 输出转换速率可配置为快或慢
- 输出驱动强度可配置为 4mA/8mA/12mA/16mA

8.2.5 模拟函数

当 GPIO 引脚配置为模拟函数时：

- 禁用输出
- 禁用输入
- 禁用上拉和下拉
- 对输入数据寄存器的读访问返回 0
- 如果 IO 用于多个模拟函数，则只能同时启用一个函数

8.2.6 I/O 数据按位处理

对于 GPIO_DATAOUT 中的每个位，对应 GPIO_bitsET 寄存器和 GPIO_bit CLR 寄存器中的两个控制位。当写 1 到 GPIO_bitsET[i]时，设置对应的 DATAOUT[i]位。当写 1 到 GPIO_bit CLR[i]时，复位对应的 DATAOUT[i]位。

将 GPIO_bitsET[i]或 GPIO_BICLR[i]中的任何位写入 0 对 GPIO_DATAOUT 中的相应位没有任何影响。如果试图同时设置和重置 GPIO_bitset 和 GPIO_bit CLR 中的位，则设置操作优先。

从 GPIO_DATAOUT 读取，GPIO_bitsET 或 GPIO_bit CLR 都返回当前的 DATAOUT 值。

逐位处理的目的是允许对任何 GPIO_DATAOUT 寄存器进行原子读/修改访问。这样，在读取访问和修改访问之间就不会有发生 IRQ 的风险。

8.3 寄存器

表 19 GPIO 寄存器

偏移	首字母缩写	寄存器描述
00h	GPIO_DATAIN	GPIO 数据输入寄存器
04h	GPIO_DATAOU	GPIO 数据输出寄存器
08h	GPIO_bitsET	GPIO 数据输出位设置寄存器
0Ch	GPIO_位 CLR	GPIO 数据输出位清除寄存器
10h	GPIO_OE	GPIO 输出启用寄存器
14h	GPIO_IE	GPIO 输入启用寄存器
18h	GPIO_PU	GPIO 上拉寄存器
1Ch	GPIO_PD	GPIO 下拉寄存器
20h	GPIO_CS	GPIO CMOS/施密特输入类型寄存器
24h	GPIO_SL	GPIO 回转速率寄存器
28h	GPIO_PDRV0	GPIO 输出驱动强度 0 寄存器
2Ch	GPIO_PDRV1	GPIO 输出驱动强度 1 寄存器
30h	GPIO_ODEN	GPIO 开漏启能寄存器
34h	GPIO_ALTFL	GPIO0~15 替代函数选择寄存器
38h	GPIO_ALTFH	GPIO16~24 替代函数选择寄存器
3Ch	GPIO_ANAE	GPIO 模拟通道启用寄存器

偏移地址： 03-00h

GPIO 数据输入寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	GPIO_DATAIN	RO	0	GPIO 输入数据值

偏移地址： 07-04h

GPIO 数据输出寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	GPIO_DATAOUT	RW	0	GPIO 输出数据值

偏移地址： 0B-08h

GPIO 数据输出位设置寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	GPIO_位 SET	RW1S	0	一次写 1 设置输出数据值； 写 0 没有任何影响 从寄存器中读取返回输出数据值

偏移地址： 0F-0Ch

GPIO 数据输出位清除寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	GPIO_位 CLR	RW1C	0	一次写 1 清输出数据值； 写 0 没有任何影响 从寄存器中读取返回输出数据值

偏移地址： 13-10h

GPIO 输出启能寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	GPIO_OE	RW	0	GPIO 输出启用位 0： 禁用 1： 启用

偏移地址： 17-14h

GPIO 输入启能寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的

23：0	GPIO_IE	RW	0	GPIO 输入启用位 0： 禁用 1： 启用

偏移地址： 1B-18h

GPIO 上拉寄存器

位	字段名	属性	默认	字段描述
31：26	-	RO	0	预留的
25：0	GPIO_PU	RW	0000002h	位 25： 启用引导 1 引脚上拉 位 24： 启用引导 0 引脚上拉 位 23-0： 启用 GPIO23~0 引脚上拉

偏移地址： 1F-1Ch

GPIO 下拉寄存器

位	字段名	属性	默认	字段描述
31：26	-	RO	0	预留的
25：0	GPIO_PD	RW	3000001h	位 25： 启用引导 1 引脚下拉 位 24： 启用引导 0 引脚下拉 位 23-0： 启用 GPIO23~0 引脚下拉

偏移地址： 23-20h

GPIO CMOS/施密特输入类型寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	GPIO_CS	RW	0	GPIO 输入类型选择 0： CMOS 缓存 1： 施密特触发器

偏移地址： 27-24h

GPIO 回转速率寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	GPIO_SL	RW	0	GPIO 输出转换速率选择 0： 快 1： 慢

偏移地址： 2B-28h

GPIO 输出驱动强度 0 寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	GPIO_PDRV0	RW	0	GPIO 输出驱动器强度 0 选择，{GPIO_PDRV1，GPIO_PDRV0} 00： 4mA 01： 8mA 10： 12mA 11： 16mA

偏移地址： 2F-2Ch

GPIO 输出驱动强度 1 寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	GPIO_PDRV1	RW	0	GPIO 输出驱动器强度 1 选择，{GPIO_PDRV1，GPIO_PDRV0} 00： 4mA 01： 8mA 10： 12mA 11： 16mA

偏移地址： 33-30h

GPIO 开漏启能寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	GPIO_ODEN	RW	0	GPIO 输出模式选择 0：推挽模式 1： 开漏模式

偏移地址： 37-34h

GPIO0~15 替代函数选择寄存器

位	字段名	属性	默认	字段描述
31：30	GPIO15_ALTF	RW	0	GPIO15 替代函数选择 00： GPIO15 模拟函数 01： UART1_CTS_N 10： SPI0_NSS3 11： 模拟函数
29：28	GPIO14_ALTF	RW	0	GPIO14 替代函数选择 00： GPIO14 01： UART1_RTS_N 10： SPI0_NSS2 11： 模拟函数
27：26	GPIO13_ALTF	RW	0	GPIO13 替代函数选择 00： GPIO13 01： UART1_TXD 10： SPI0_NSS1 11： 模拟函数

25：24	GPIO12_ALTF	RW	0	GPIO12 替代函数选择 00： GPIO12 01： UART1_RXD 10： MCO 11： 模拟函数
23：22	GPIO11_ALTF	RW	0	GPIO11 替代函数选择 00： GPIO11 01： SPI0_NSS0 10： RTC_1Hz 11： 模拟函数
21：20	GPIO10_ALTF	RW	0	GPIO10 替代函数选择 00： GPIO10 01： SPI0_MISO 10： 预留的 11： 模拟函数
19：18	GPIO9_ALTF	RW	0	GPIO9 替代函数选择 00： GPIO9 01： SPI0_MOSI 10： I2C1_SDA 11： 模拟函数
17：16	GPIO8_ALTF	RW	0	GPIO8 替代函数选择 00： GPIO8 01： SPI0_SCK 10： I2C1_SCL 11： 模拟函数

15：14	GPIO7_ALTF	RW	0	GPIO7 替代函数选择 00： GPIO7 01： I2C0_SDA 10： 预留的 11： 模拟函数
13：12	GPIO6_ALTF	RW	0	GPIO6 替代函数选择 00： GPIO6 01： I2C0_SCL 10： 预留的 11： 模拟函数
11：10	GPIO5_ALTF	RW	0	GPIO5 替代函数选择 00： GPIO5 01： UART0_CTS_N 10： 预留的 11： 模拟函数
9：8	GPIO4_ALTF	RW	0	GPIO4 替代函数选择 00： GPIO4 01： UART0_RTS_N 10： SPI1_NSS3 11： 模拟函数
7：6	GPIO3_ALTF	RW	0	GPIO3 替代函数选择 00： GPIO3 01： UART0_TXD 10： SPI1_NSS2 11： 模拟函数

5：4	GPIO2_ALTF	RW	0	GPIO2 替代函数选择 00： GPIO2 01： UART0_RXD 10： SPI1_NSS1 11： 模拟函数
3：2	GPIO1_ALTF	RW	0	GPIO1 替代函数选择 00： SWDIO 01： GPIO1 10： 预留的 11： 模拟函数
1：0	GPIO0_ALTF	RW	0	GPIO0 替代函数选择 00： SWCLK 01： GPIO0 10： HSE_CLK 11： 模拟函数

偏移地址： 3B-38h

GPIO16~23 替代函数选择寄存器

位	字段名	属性	默认	字段描述
31：16	-	RO	0	预留的
15：14	GPIO23_ALTF	RW	0	GPIO23 替代函数选择 00： GPIO23 01： TIMER1_EXTIN 10： 预留的 11： 模拟函数

13：12	GPIO22_ALTF	RW	0	GPIO22 替代函数选择 00： GPIO22 01： TIMER0_EXTIN 10： 预留的 11： 模拟函数
11：10	GPIO21_ALTF	RW	0	GPIO21 替代函数选择 00： GPIO21 01： PWM_OUT6 10： COMP1_OUT 11： 模拟函数
9：8	GPIO20_ALTF	RW	0	GPIO20 替代函数选择 00： GPIO20 01： PWM_OUT5 10： COMP0_OUT 11： 模拟函数
7：6	GPIO19_ALTF	RW	0	GPIO19 替代函数选择 00： GPIO19 01： PWM_OUT4 10： SPI1_NSS0 11： 模拟函数
5：4	GPIO18_ALTF	RW	0	GPIO18 替代函数选择 00： GPIO18 01： PWM_OUT3 10： SPI1_MISO 11： 模拟函数

3：2	GPIO17_ALTF	RW	0	GPIO17 替代函数选择 00： GPIO17 01： PWM_OUT2 10： SPI1_MOSI 11： 模拟函数
1：0	GPIO16_ALTF	RW	0	GPIO16 替代函数选择 00： GPIO16 01： PWM_OUT1 10： SPI1_SCK 11： 模拟函数

偏移地址： 3F-3Ch

GPIO 模拟通道启用寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：0	GPIO_ANAE	RW	0	GPIO 模拟通道启用位 0：禁用模拟通道 1：启用模拟通道

9 扩展中断和事件控制器（EXTI）

9.1 概述

扩展中断和事件控制器(EXTI)通过可配置的直接事件输入(行)管理 CPU 和系统唤醒。它向电源控制提供唤醒请求，并向 CPU NVIC 生成中断请求，并向 CPU 事件输入生成事件。

EXTI 唤醒请求允许从停止模式唤醒系统。

中断请求和事件请求生成也可以在运行模式中使用。

9.1.1 功能列表

- 在任何输入的事件上的系统唤醒
- 在源外设中没有唤醒标志的事件的唤醒标志和 CPU 中断生成
- 可配置事件(来自 I/O，外设没有相关的中断挂起状态位)
  - 可选择主动触发边缘
  - 独立的升降边缘中断挂起状态位
  - 单个中断和事件生成屏蔽，用于调节 CPU 唤醒，中断和事件生成
  - SW 触发可能性
- 直接事件(来自具有关联标志和中断挂起状态位的外设)
  - 固定上升沿主动触发
  - EXTI 中没有中断挂起状态位
  - 单独的中断和事件生成屏蔽，用于调节 CPU 唤醒和事件生成
  - 无 SW 触发可能性

9.2 函数描述

9.2.1 EXTI 连接

当系统处于停止模式时，能够生成唤醒或中断事件的外设连接到 EXTI。

- 产生脉冲或在外设中没有中断状态位的外设唤醒信号连接到 EXTI 可配置线。对于这些事件，EXTI 提供了一个需要清除的状态挂起位。 与状态位相关联的 EXTI 中断中断了 CPU。
- 在外设中具有需要在外设中清除的状态位的外设中断和唤醒信号连接到 EXTI 直线。EXTI 中没有状态挂起位。中断或唤醒由外设的 CPU 清除。直接中断 CPU 的是外设中断。

EXTI 可配置的事件中断连接到 CPU 的 NVIC。

专用的 EXTI CPU 事件连接到 CPU rxev 输入。

EXTI CPU 唤醒信号连接到 PMU 块，用于唤醒系统和 CPU 子系统总线时钟。

表 20 EXTI 线连接

EXTI 线	线源	线类型
0~23	GPIO	可配置
24	LVD 输出	可配置
25	COMP0 输出	可配置
26	COMP1 输出	可配置
27	Charge_ok	可配置
28	Charge_end	可配置
29	过温	可配置
30	Rtc_报警	直接
31	Rtc_wut	直接

9.2.2 EXTI 可配置事件输入唤醒

软件中断事件寄存器允许由软件触发可配置的事件，并写入相应的寄存器位，而不考虑边缘选择设置。

上升沿和下降沿选择寄存器允许启用和选择可配置事件主动触发边缘或两个边缘。

CPU 有专用的中断屏蔽寄存器和专用的事件屏蔽寄存器。启用事件功能后，CPU 会产生一个事件。一个 CPU 的所有事件都被 OR-ed 一起转换成一个 CPU 事件信号。未为未屏蔽的 CPU 事件设置事件未挂起寄存器（EXTI_RPR 和 EXTI_FPR）。

可配置事件具有唯一的中断挂起请求寄存器，由 CPU 共享。挂起的寄存器只为一个未被屏蔽的中断而设置。每个可配置事件向 CPU 提供一个公共中断。可配置的事件中断需要由软件在 EXTI_RPR 和/或 EXTI_FPR 寄存器中确认。

当 CPU 中断或 CPU 事件被启用时，异步边缘检测电路被时钟延迟和上升边检测脉冲发生器复位。这保证了在异步边缘检测电路复位之前唤醒 EXTI fclk 时钟。

对于可配置的事件输入，软件可以通过设置软件中断/事件寄存器 EXTI_SWIER 的相应位来生成事件请求，该位对事件输入具有上升边的作用。挂起上升边事件标志设置在 EXTI_RPR 寄存器中，与 EXTI_RTSR 寄存器设置无关。

9.2.3 EXTI 直接事件输入唤醒

直接事件没有关联的 EXTI 中断。EXTI 只唤醒系统和 CPU 子系统时钟，并可能生成一个 CPU 唤醒事件。与直接唤醒事件相关联的外设同步中断唤醒 CPU。

EXTI 直接事件能够生成 CPU 事件。这个 CPU 事件唤醒了 CPU。CPU 事件可能在相关外设的中断标志位设置之前发生。

对于直接事件输入，当在相关外设中启用时，仅在上升沿上生成事件请求。EXTI 中没有相应的 CPU 挂起位。当相关的 CPU 中断被解除屏蔽时，相应的 CPU 子系统将被唤醒。CPU 被外设同步中断唤醒(中断)。

9.3 寄存器

表 21 EXTI 寄存器

偏移	首字母缩写	寄存器描述
00h	EXTI_RTSR	EXTI 上升触发选择寄存器
04h	EXIT_FTSR	EXTI 下降触发选择寄存器
08h	EXTI_SWIER	EXTI 软件中断事件寄存器
0Ch	EXTI_RPR	EXTI 上升沿挂起寄存器
10h	EXTI_FPR	EXTI 下降沿挂起寄存器
14h	EXTI_IMR	EXTI CPU 使用中断屏蔽寄存器唤醒
18h	EXTI_EMR	EXTI CPU 使用事件屏蔽寄存器唤醒

偏移地址： 03-00h

EXTI 上升触发选择寄存器

位	字段名	属性	默认	字段描述
31：30	-	RO	0	预留的
29：0	RISE_TRGEN	RW	0	可配置行 x 的上升触发事件配置位(x = 29 到 0) 每个位启用/禁用相应行上的事件和中断的上升边触发器 0： 禁用 1： 启用

偏移地址： 07-04h

EXTI 下降触发选择寄存器

位	字段名	属性	默认	字段描述
31：30	-	RO	0	预留的

29：0	FALL_TRGEN	RW	0	可配置行 x 的下降触发事件配置位(x = 29 到 0) 每个位在相应的行上启用/禁用事件和中断的下降边触发器 0： 禁用 1： 启用

偏移地址： 0B-08h

EXTI 软件中断事件寄存器

位	字段名	属性	默认	字段描述
31：30	-	RO	0	预留的
29：0	SW_TRGEN	WO	0	软件上升触发事件触发 x 行(x = 29 到 0) 软件对任何位的设置都会在相应的行 x 上触发上升边事件，导致中断，与 EXTI_RTSR 和 EXTI_FTSR 设置无关。这些位由 HW 自动清除。读取任何位总是返回 0。 0： 无作用 1： 在相应的行上生成的上升的边缘事件，随后是一个中断

偏移地址： 0F-0Ch

EXTI 上升沿挂起寄存器

位	字段名	属性	默认	字段描述
31：30	-	RO	0	预留的
29：0	RISE_PEND	RW1C	0	可配置行 x 的上升沿事件（x = 29 到 0） 每个位都是由相应行上的硬件或软件生成的上升沿事件设置的。每个位都通过写入 1 来清除。 0： 未发生上升沿触发请求 1：发生上升沿触发请求

偏移地址： 13-10h

EXTI 下降沿挂起寄存器

位	字段名	属性	默认	字段描述
31：30	-	RO	0	预留的
29：0	FALL_PEND	RW1C	0	可配置行 x 的下降沿事件（x = 29 到 0） 每个位都是由相应行上的硬件或软件生成的下降沿事件设置的，每个位都通过写入 1 来清除。 0： 未发生下降沿触发请求 1： 发生下降沿触发请求

偏移地址： 17-14h

具有中断屏蔽寄存器的 EXTI CPU 唤醒

位	字段名	属性	默认	字段描述
31：0	INT_MASK	RW	0	第 x 行带有中断屏蔽的 CPU 唤醒（x = 31 到 0） 设置/清除每个位，通过相应行上的事件中断解除/屏蔽 CPU 唤醒。 0： 用中断屏蔽唤醒 1： 在中断解除屏蔽的情况下唤醒

偏移地址： 1B-18h

具有事件屏蔽寄存器的 EXTI CPU 唤醒

位	字段名	属性	默认	字段描述
31：0	EVT_MASK	RW	0	第 x 行上带有事件生成屏蔽的 CPU 唤醒(x = 31 到 0) 设置/清除每个位，通过相应行上的事件生成解除/屏蔽 CPU 唤醒。 0：已屏蔽事件生成的唤醒 1： 解除屏蔽事件生成的唤醒

10 串行外设接口（SPI）

10.1 概述

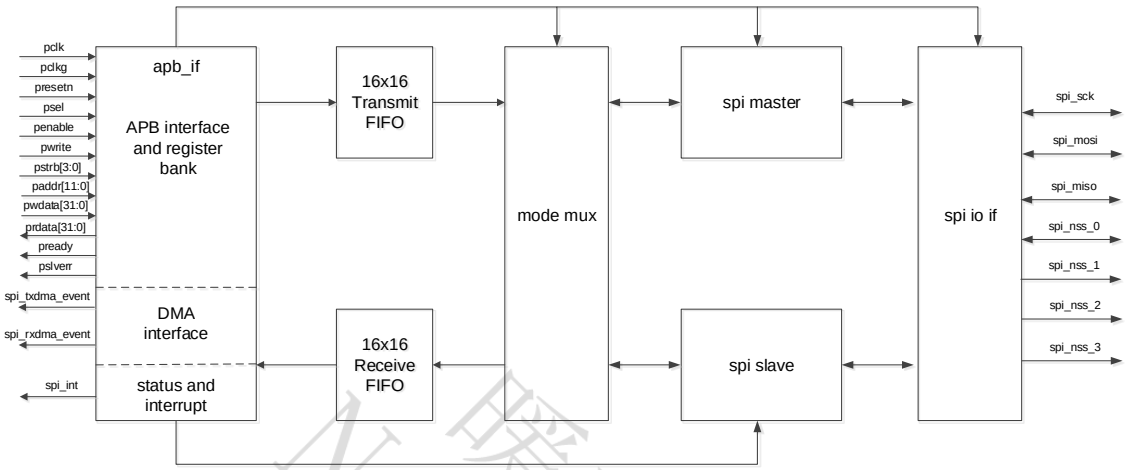
串行外围接口（SPI）是高速同步串行输入/输出端口，允许编程长度（4 至 16 位）的串行位流以编程位传输速率进出设备。SPI 通常用于设备与外部外设之间的通信。典型的应用包括通过移位寄存器、显示驱动程序、SPI EPROMS 和模数转换器等设备进行外部 I/O 或外设扩展的接口。

10.1.1 功能列表

- 主从模式
- 三条线路上的全双工同步传输
- 双线半双工同步传输(带双向数据线)
- 双线单工同步传输(单向数据线)
- 发送/接收 16x16 位 FIFO
- 4-16 位数据大小选择
- 主模式波特率发生器高达 Fpclk/2
- 从模式波特率发生器高达 Fpclk/4
- 多从芯片选择（nss0~nss3）
- 通过硬件或软件进行 NSS 管理
- 可编程时钟极性和相位
- 可编程的数据顺序与 MSB 或 LSB 移位
- 中断能力
- DMA 事件支持

10.2 框图

图 9 SPI 框图



- spi_apb_if 作为 apb 从属设备，可以生成寄存器、dma 事件和中断。
- 两个 16x16 位传输和接收 FIFOs，用于缓冲输出或输入数据。
- spi_mode_mux 合并 spi 主和 spi 从数据或至 FIFOs/来自 FIFOs 的请求。
- spi 主设备负责协议的实现，包括 tx 方向的并-串行转换器，rx 方向的串行并行转换器，基于相位、极性和波特率的 sck 生成。
- spi 从属设备负责协议的实现，包括 tx 方向的并行到串行转换器、rx 方向的串行并行转换器、基于相位和极性的 rx 数据采样。
- Spi_io_if 在主设备和从设备之间执行 pinmux 操作。

10.3 函数描述

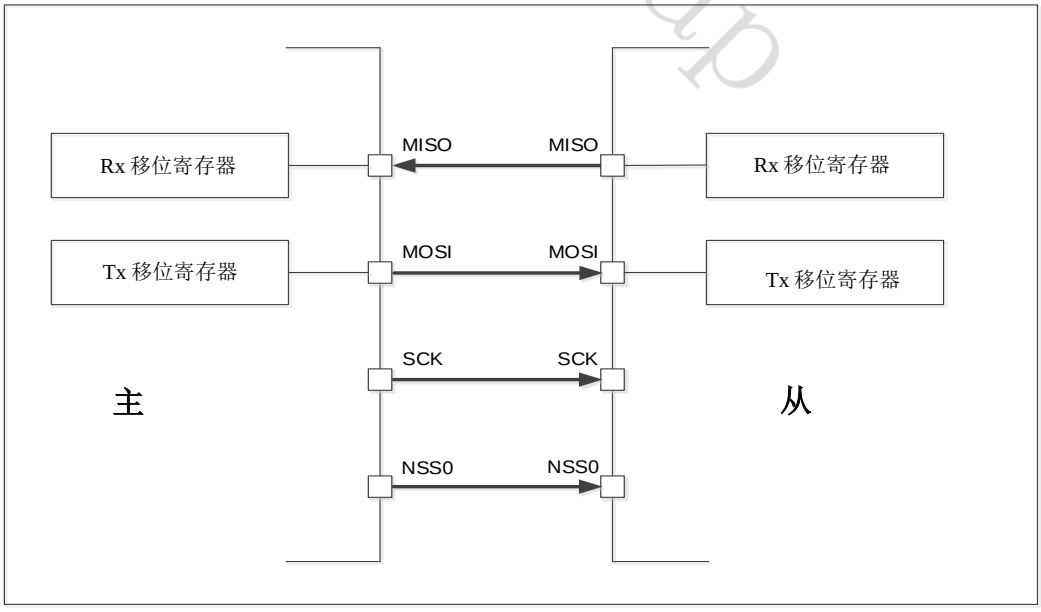
10.3.1 SPI 引脚

- **MISO：** 主输入/从输出数据，一般情况下，该引脚用于从模式下传输数据和主模式下接收数据。
- **MOSI：** 主输出/从输入数据，一般情况下，该引脚用于主模式下的数据传输和从模式下的数据接收。
- **SCK：** SPI 主设备的串行时钟输出引脚或 SPI 从属设备的的输入引脚。
- **NSS0：** 从 0 选择引脚，串行外设接口主设备输出或串行外设接口从设备输入引脚。
- **NSS1：** 从 1 选择引脚，串行外设接口主设备输出。
- **NSS2：** 从 2 选择引脚，串行外设接口主设备输出。
- **NSS3：** 从 3 选择引脚，串行外设接口主设备输出。

10.3.2 通信

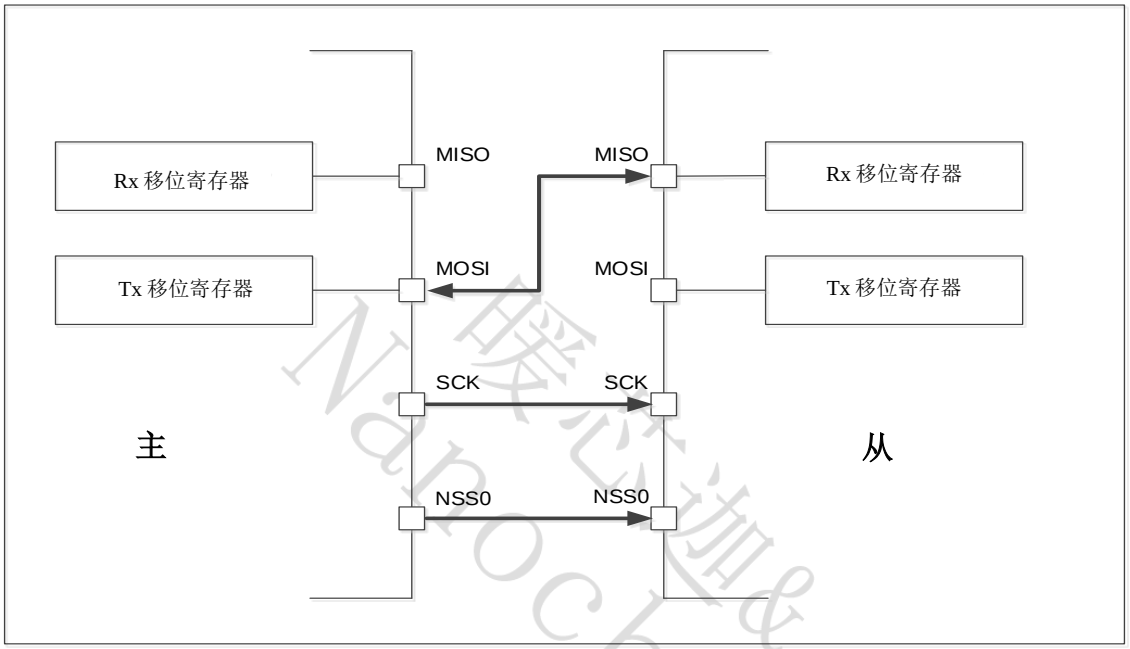
默认情况下，SPI 配置为全双工通信。在这种配置中，主和从的移位寄存器使用 MOSI 和 MISO 引脚之间的两条单向线连接。 在 SPI 通信过程中，数据在主机提供的 SCK 时钟边缘上同步移位。主设备通过 MOSI 线将数据发送给从设备，并通过 MISO 线接收从设备的数据。 当数据帧传输完成(所有位都被移位)，主从机之间的信息就交换了。

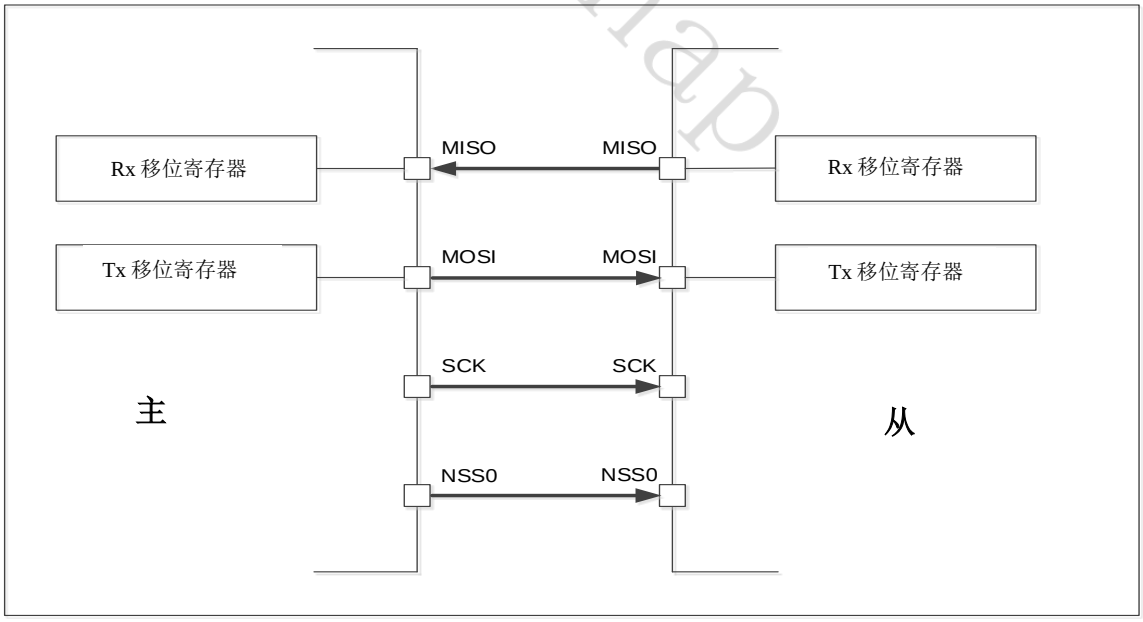
图 10 全双工的应用程序



SPI 可以通过在 SPI_CTRL1 寄存器中设置 BIDI_EN 以半双工模式进行通信。 在这种配置中，使用一条交叉连接线将主从机的移位寄存器连接在一起。在此通信期间，数据在 SCK 时钟边缘上的移位寄存器之间同步移动，传输方向由主寄存器和从寄存器使用 SPI_CTRL1 寄存器中的 BIDI_MODE 位相互选择。在此配置中，主设备的 MISO 引脚和从设备的 MOSI 引脚可自由用作 GPIO 的其他应用程序。

图 11 半双工传输应用





通过使用 SPI_CTRL1 寄存器中的 UNIDI_MODE 位将 SPI 设置为仅发送或仅接收，SPI 可以以单工模式进行通信。在这种配置中，只有一条线用于主和从设备的移位寄存器之间的传输。其余的 MISO 和 MOSI 引脚对不用于通信，可作为标准 GPIOs。

图 12 主模式为发送模式，从模式为接收模式

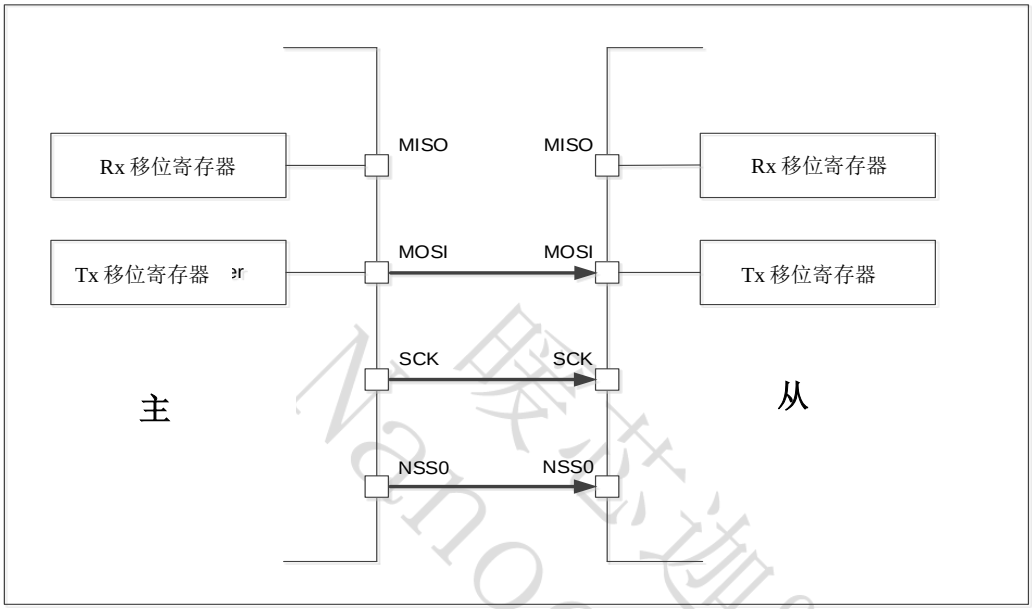
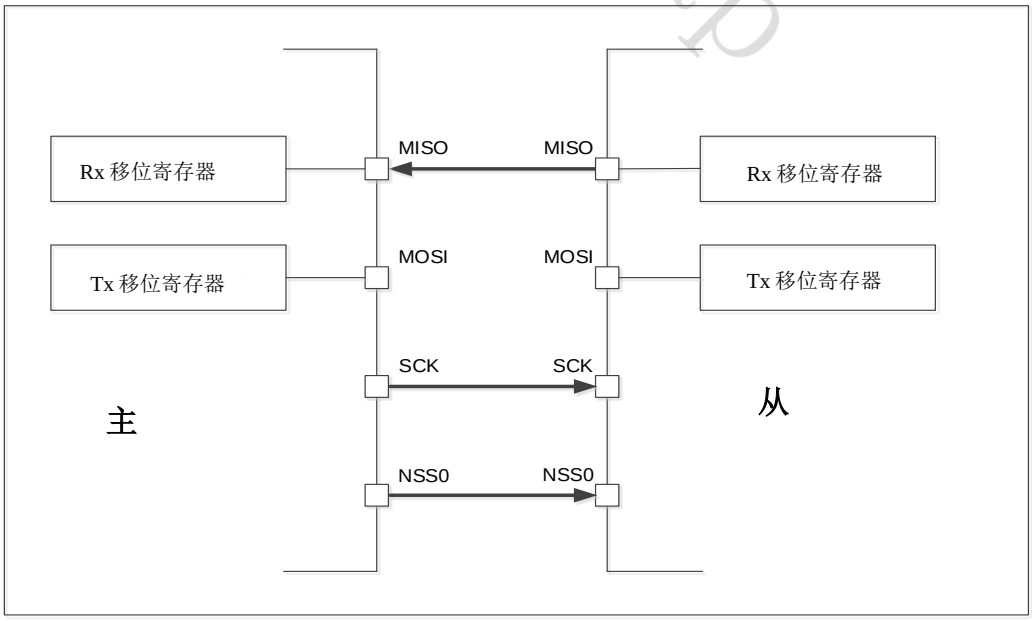
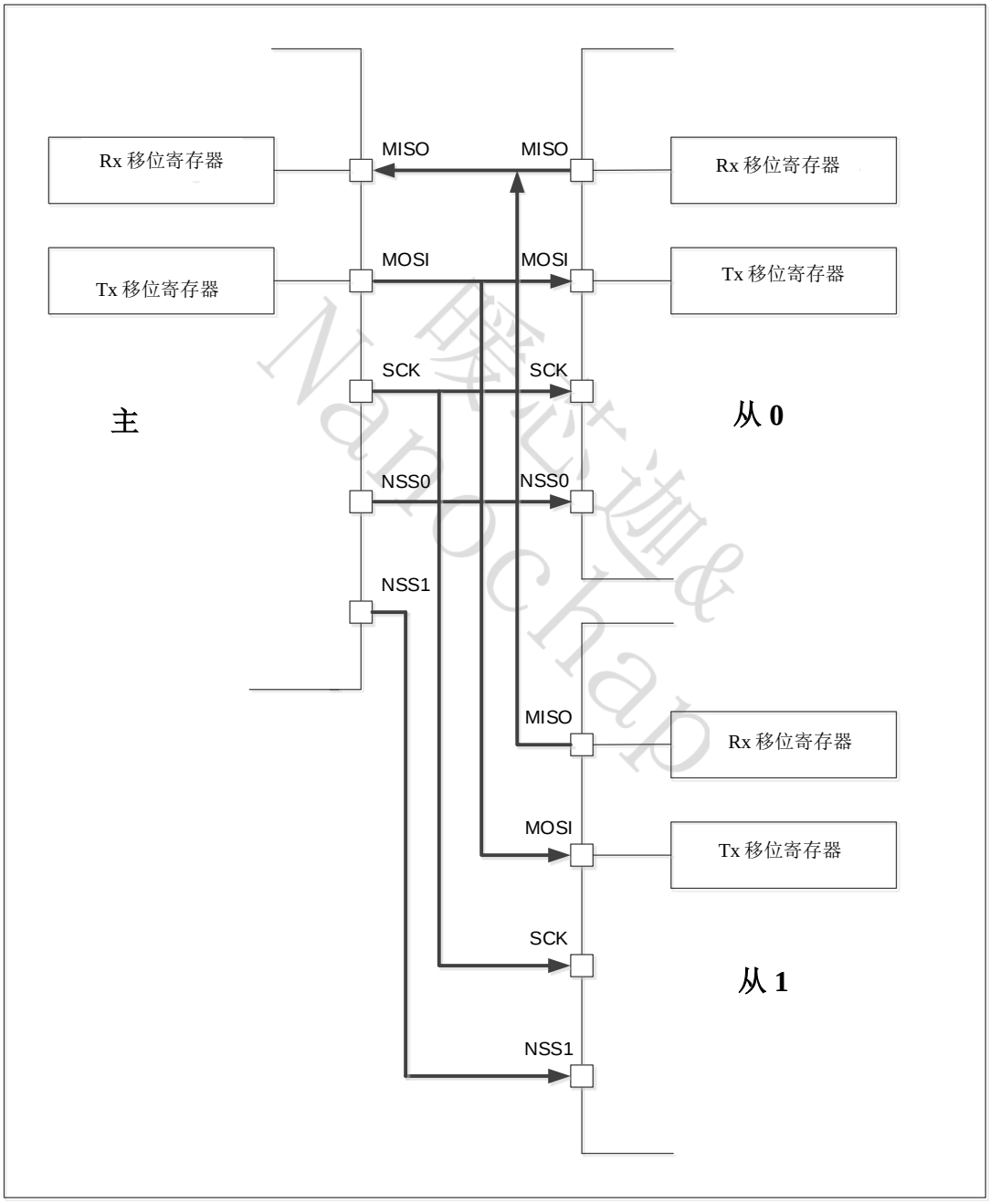


图 13 主模式为接收模式，从模式为发送模式



在具有多达 4 个独立从设备的配置中，主设备使用 nss0~nss3 引脚来管理每个从设备的芯片选择线。主设备可以通过在 SPI_CTRL2 寄存器中设置 NSS0_EN~NSS3_EN 位来选择一个从设备。完成此操作后，将建立一个标准的主从通信。

图 14 主多从应用程序

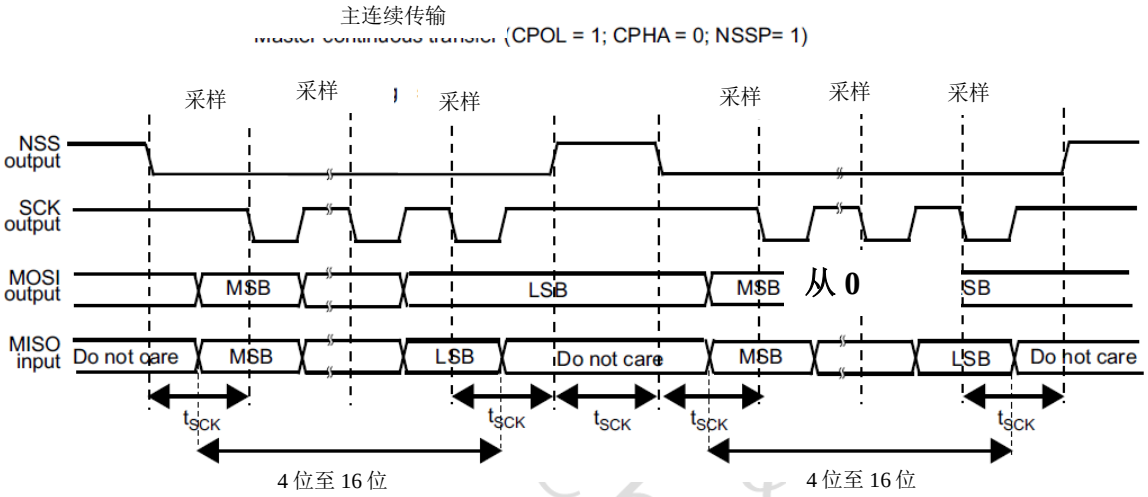


在从模式下，NSS 作为一个标准的芯片选择输入，让从与主设备通信。

在主模式下，NSS 被用作输出引脚。当 SPI_CTRL1 寄存器中的 NSS_MST_CTRL bit 过高时，由 SW 设置 NSS_MST_SW bit 驱动 NSS。当 NSS_MST_CTRL 位低时，NSS 由 HW 自动驱动。

NSS 脉冲模式由 SPI Control1 寄存器中的 NSS_TOGGLE 激活。当激活时，当 NSS 至少在一个时钟周期的持续时间内保持在高电平时，在两个连续的数据帧传输之间产生一个 NSS 脉冲。这种模式允许从设备锁住数据。

图 15 NSS 脉冲模式



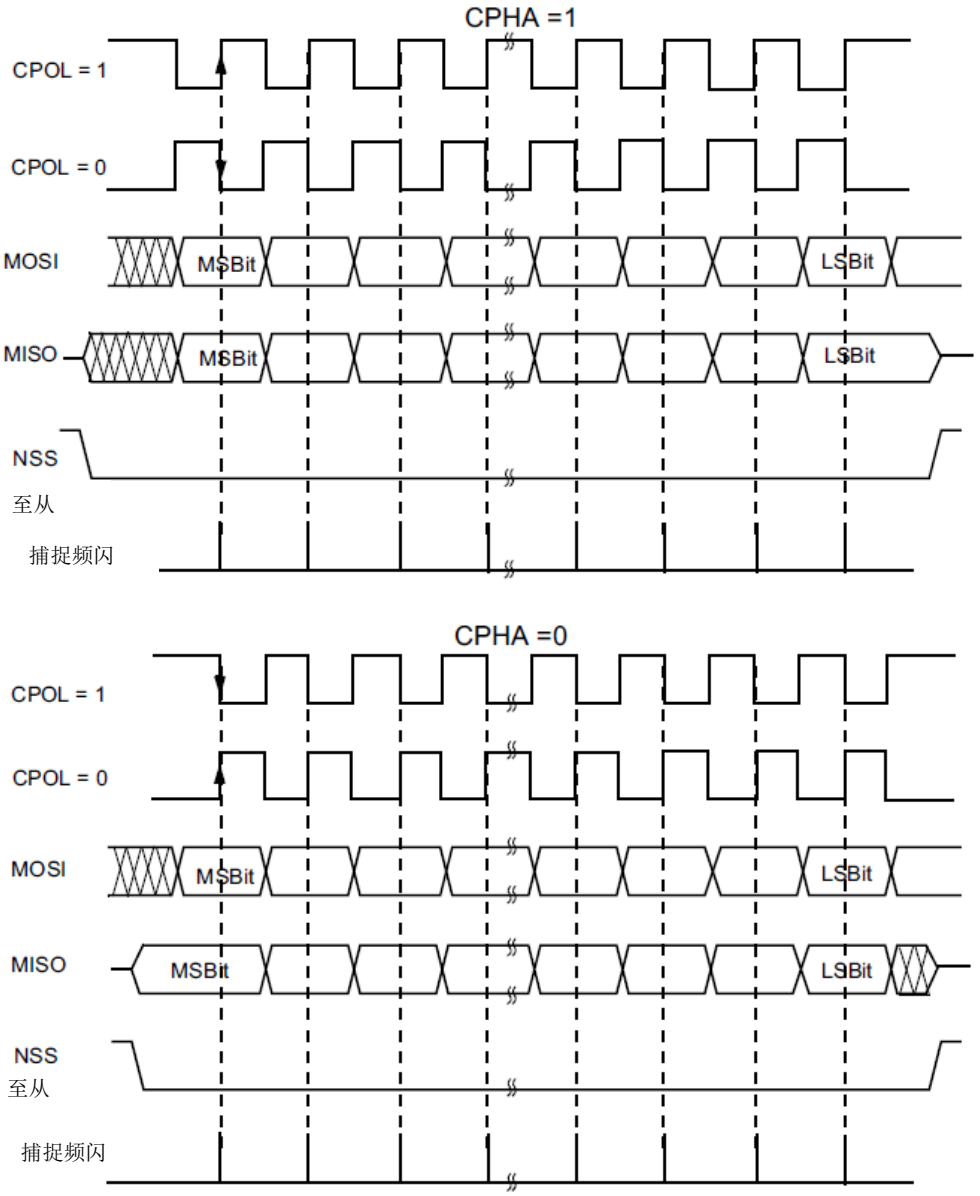
在 SPI 通信过程中，接收和发送操作是同时进行的。串行时钟(SCK)同步数据线上信息的移位和采样。通信格式取决于时钟相位、时钟极性和数据帧格式。为了能够一起通信，主设备和从设备必须遵循相同的通信格式。

软件可以使用 SPI_CTRL1 寄存器中的 CPOL 和 CPHA 位来选择四种可能的定时关系。时钟极性位控制无数据传输时时钟的空闲状态值。这个位同时影响主模式和从模式。如果 CPOL 复位，则 SCK 引脚处于低电平空闲状态。如果设置了 CPOL，则 SCK 引脚处于高电平空闲状态。

如果设置了 CPHA 位，SCK 引脚上的第二条边捕获交易的第一个数据位(如果 CPOL 位被重置，则下降沿，如果 CPOL 位被设置，则上升沿)。每次出现这种时钟转换类型时，都会锁定数据。如果 CPHA 位重置，SCK 引脚上的第一个边捕获交易的第一个数据位(如果 CPOL 位设置为下降沿，如果 CPOL 位重置为上升沿)。每次出现这种时钟转换类型时，都会锁定数据。

CPOL 位和 CPHA 位的组合选择数据捕获时钟边缘。

图 16 数据时钟时序图



SPI 移位寄存器可以设置为 MSB-first 或 LSB-first 移位，这取决于 SPI_CTRL1 寄存器中的 LSB_SEL 位的值。通过使用 SPI_CTRL2 寄存器中的 CHAR_LEN 位来选择数据帧大小。它可以设置从 1 位到 16 位的长度，该设置适用于传输和接收。

10.3.3 SPI 配置

主用和从用的配置过程几乎相同。初始化标准通信时，执行以下步骤：

1) 写正确的 GPIO 改变寄存器：为 MOSI，MISO,SCK 和 NSS 引脚配置 GPIO。

2) 写入 SPI_CTRL1 寄存器：

a. 使用 BAUD_RATE[2：0]位配置串口时钟波特率。

b. 配置 CPOL 和 CPHA 位组合，定义数据传输与串行时钟之间的四种关系之一。

c. 通过配置 BIDI_EN、BIDI_MODE、UNIDI_MODE bits，选择全双工、半双工或单工通信方式。

d. 配置 LSB_SEL 位定义帧格式。

e. 通过配置 NSS_TOGGLE、NSS_MST_CTRL、NSS_MST_SW bits 选择 NSS 控制方式。

f. 通过配置 MST_SLV_SEL 位选择主模式或从模式。

3) 写入 SPI_CTRL2 寄存器：

a. 配置 CHAR_LEN[3：0]位来选择传输的数据长度。

b. 通过配置 NSS0_EN、NSS1_EN、NSS2_EN 或 NSS3_EN 位，选择“NSS 端口”。

c. 通过配置 SAMP_PHASE[1：0]位选择正确的主设备 RX 数据采样阶段。

d. 根据从机需求，配置 C2T_DELAY 位和 T2C_DELAY 位，选择合适的 C2T/T2C 时延。

e. 通过配置 TXDMA_EN 和 RXDMA_EN 位，在 FIFO 模式下启用或禁用 TX/RX DMA。

4) 写入 FIFO 控制寄存器：

a. 配置 TX_FIFO_TH 或 RX_FIFO_TH 来定义触发级别阈值。

b. 通过配置 TX_FIFO_CLR 位和 RX_FIFO_CLR 位清除 TX/RX FIFO。

c. 通过配置 FIFO_EN 位，启用或禁用 FIFO 模式。

10.3.4 序列处理

当传输被启用时，一个序列开始并继续，而任何数据都存于主设备的 TXFIFO 中。时钟信号由主设备连续提供，直到 TXFIFO 为空，然后停止等待其他数据。

在仅接收模式中，当两个 SPI 都被启用并且仅接收模式被激活时，主设备立即启动序列。时钟信号由主控程序提供，直到主设备禁用 SPO 或仅接收模式，它才会停止。到目前为止，主设备连续接收数据帧。

虽然主设备可以以连续模式提供所有事务，但它必须尊重从设备随时处理数据流及其内容的能力。必要时，主机必须减慢通信速度，并提供一个较慢的时钟或具有足够延迟的独立帧或数据会话。主设备没有下溢错误，只有溢出错误。对于从设备，同时存在下溢和溢出错误。

每个序列必须被 NSS 脉冲封装与多从机系统平行为通信选择仅仅一个从机。在单从系统中，不需要用 NSS 来控制从系统，但通常更好的方法是在这里提供脉冲，使从系统与每个数据序列的开始同步。NSS 可以通过软件和硬件两种方式进行管理。

■ **启用 SPI 的步骤：**

当设置了 SPI_BUSY 位时，表示正在进行的数据帧事务。当专用帧事务完成时，会引发 RXNE 标志。

建议在主设备发送时钟之前启用 SPI 从设备。否则，可能会出现不期望的数据传输。在开始与主设备通信之前，从机的数据寄存器必须已经包含要发送的数据。在 SPI 从设备启用之前，SCK 信号必须设定在与所选极性相对应的空闲状态水平。

当 SPI 被启用且 TXFIFO 不为空时，或者当下一次写入 TXFIFO 时，主设备在全双工(或任何仅传输模式)下开始通信。

在任何主设备接收模式下，主设备开始通信，时钟在 SPI 启用后立即开始运行。

■ **禁用 SPI 的步骤：**

当禁用 SPI 时，必须遵循禁用过程。在系统进入低功耗模式之前执行此操作是很重要的。当外设时钟停止时，在这种情况下正在进行的事务可能会损坏。在某些模式下，禁用过程是停止持续通信运行的唯一方法。

全双工模式或仅传输模式的主设备在停止提供数据传输时可以完成任何事务。在这种情况下，时钟在最后一个数据事务结束后停止。

当主设备处于任意接收模式时，停止连续时钟的唯一方法是禁用 SPI 这必须发生在最后一个数据帧事务的特定时间窗口内，恰好在其第一个位的采样时间到其最后一个比特传输开始之前(为了接收完整数量的预期数据帧，并防止在最后一个有效数据帧之后出现任何额外的“假”数据读取)。在这种模式下禁用 SPI 时必须遵循特定的程序。

当禁用 SPI 时，接收但未读取的数据仍然存储在 RXFIFO 中，并且必须在下次启用 SPI 时进行处理，然后再开始新的序列。要防止有未读数据，请使用正确的禁用过程，确保禁用 SPI 时 RXFIFO 为空。

标准的禁用过程是基于在 FIFO 状态寄存器中提取 SPI_BUSY 状态和 TX_FIFO_LEN[4：0]来检查传输会话是否完全完成。

正确的禁用过程是(除非使用接收模式)：

1) 等待 TX_FIFO_LEN[4：0]= 5'd0(没有更多的数据传输)。

2) 等待直到 SPI_BUSY=0(处理最后一个数据帧)。

3) 通过配置 SPI_EN=0 禁用 SPI。

4) 读取数据直至 RX_FIFO_LEN[4：0]=5’d0 (读取所有接收数据)。

对于某些接收模式，正确的禁用过程是：

1) 在最后一个数据帧正在进行时，通过在特定的时间窗口禁用 SPI 来中断接收流(在 FIFO 模式下，可以读取 RX_FIFO_LEN，然后决定禁用 SPI。在非 FIFO 模式下，可以等待溢出错误，然后禁用 SPI)。

2) 等待直至 SPI_BUSY=0 (处理最后一个数据帧)。

3) 读取数据直至 RX_FIFO_LEN[4：0]=5’d0 (读取所有接收数据)。

- **忙碌标志：**

SPI_BUSY 标志由硬件设置和清除。

当设置 SPI_BUSY 时，表示 SPI 上有数据传输正在进行(SPI 总线繁忙)。

SPI_BUSY 标志可以在某些模式下使用，以检测传输的结束，以便软件可以在进入不为外设提供时钟的低功耗模式之前禁用 SPI 或其外设时钟。

在以下任何一种情况下，SPI_BUSY 都将被清除：

1) 当正确禁用 SPI 时。

2) 在主模式下，当它完成数据传输，但没有新数据准备发送时。

3) 在从模式下，NSS 处于非激活状态。

10.4 寄存器

系统程序员可以访问和控制下列列出的任何 SPI 寄存器。这些控制 SPI 操作、接收数据和传输数据的寄存器在设备内存映射中的 32 位地址可用。

表 22 SPI 寄存器

偏移	首字母缩写	寄存器描述
00h	RBR	接收缓冲区寄存器(只读)
00h	THR	发送器保持寄存器(仅写)
04h	IER	中断启用寄存器
08h	ISR	中断状态寄存器
0Ch	CTRL1	控制 1 寄存器
10h	CTRL2	控制 2 寄存器
14h	FCR	FIFO 控制寄存器
18h	FSR	FIFO 状态寄存器
1Ch	DBG	调试信号寄存器

偏移地址： 03-00h

接收缓冲寄存器

位	字段名	属性	默认	字段描述
31：16	-	RO	0	预留的
15：0	RBR	RO	0	SPI 接收的任何数据字节都是通过读取这个寄存器来访问的

偏移地址： 03-00h

发送器保持寄存器

位	字段名	属性	默认	字段描述
31：16	-	RO	0	预留的
15：0	THR	WO	0	这个寄存器用于缓冲输出字符

偏移地址： 07-04h

中断启用寄存器

位	字段名	属性	默认	字段描述
31：5	-	RO	0	预留的
4	UNDERRUN_INT_EN	RW	0	TX 欠载中断启用位 0： UNDERRUN 中断禁用 1： UNDERRUN 中断启用
3	OVERRUN_INT_EN	RW	0	RX 超限中断启用位 0： OVERRUN 中断禁用 1： OVERRUN 中断启用
2	CMPL_INT_EN	RW	0	TX/RX 事务完成中断启用位 0： CMPL 中断禁用 1： CMPL 中断启用
1	TXE_INT_EN	RW	0	TX 缓冲空中断启用位 0： TXE 中断禁用 1： TXE 中断启用
0	RXNE_INT_EN	RW	0	RX 缓冲非空中断启用位 0： RXNE 中断禁用 1： RXNE 中断启用

偏移地址： 0B-08h

中断状态寄存器

位	字段名	属性	默认	字段描述
31：5	-	RO	0	预留的
4	UNDERRUN_INT_STS	RW1C	0	TX 欠载中断状态，可以通过对该位写入 1 来清除该位
3	OVERRUN_INT_STS	RW1C	0	RX 超限中断状态，可通过对该位写入 1 来清除该位

2	CMPL_INT_STS	RW1C	0	TX/RX 事务完成状态，当处于主模式时，当 TX FIFO 为空，RX FIFO 为满时产生此中断，当处于从模式时，传输一次数据后产生此中断，对该位写入 1 即可清除此位
1	TXE_INT_STS	RO	1	当处于非 FIFO 模式时 0： 传输缓冲区非空 1： 发送缓冲区为空，当写入 THR 寄存器时，此状态位被清除为 0 当处于 FIFO 模式时 0： TX FIFO 长度> TX FIFO 阈值 1： TX FIFO 长度<= TX FIFO 阈值
0	RXNE_INT_STS	RO	0	当处于非 FIFO 模式时 0： 接收缓冲区为空 1： 接收缓冲区非空，当读取 RBR 寄存器时，该状态位被清除为 0 当处于 FIFO 模式时 0： RX FIFO 长度 < RX FIFO 阈值 1： RX FIFO 长度>= RX FIFO 阈值

偏移地址： 0F-0Ch

SPI 控制 1 寄存器

位	字段名	属性	默认	字段描述
31：16	-	RO	0	预留的
15	BIDI_EN	RW	0	双向数据模式启用位 0：选择 2 行单向数据模式 1： 选择 1 行双向数据模式
14	BIDI_MODE	RW	0	双向模式 0： 仅传输 1： 仅接收

13：12	UNIDI_MODE	RW	0	单向模式 00： 传输与接收 01： 仅传输 10： 仅接收 11： 预留的
11	NSS_TOGGLE	RW	0	当进行连续传输时，该位允许 SPI 主程序在两个连续数据之间产生一个 NSS 脉冲 0： 未发生 NSS 脉冲 1： NSS 脉冲生成
10	LOOPBACK_EN	RW	0	内部回环测试模式，MOSI 引脚内部连接 MISO 引脚 0：禁用内部环回测试模式 1： 启用用内部环回测试模式
9	NSS_MST_SW	RW	0	由 SW 控制的 NSS 值 0： 值为 0,NSS 断言 1： 值为 1，NSS 解除断言
8	NSS_MST_CTRL	RW	0	NSS 主控 0： NSS 由 HW 控制 1： NSS 由 SW 控制
7	LSB_SEL	RW	0	帧格式 0： 首先用 MSB 发送/接收数据 1：首先用 LSB 发送/接收数据

6：4	BAUD_RATE	RW	0	波特率控制 000： Fpclk/2 001： Fpclk/4 010： Fpclk/8 011： Fpclk/16 100： Fpclk/32 101： Fpclk/64 110： Fpclk/128 111： Fpclk/256
3	CPOL	RW	0	时钟极性 0：空闲时 SCK 为 0 1： 空闲时 SCK 为 1
2	CPHA	RW	0	时钟相 0： 第一个时钟跳变是第一个数据捕获边 1： 第二个时钟跳变是第一个数据捕获边缘
1	MST_SLV_SEL	RW	1	主从模式选择 0： 从设备配置 1： 主设备配置
0	SPI_EN	RW	0	SPI 启用位 0： 禁用 SPI 1： 启用 SPI

偏移地址： 13-10h

SPI 控制 2 寄存器

位	字段名	属性	默认	字段描述
31：16	-	RO	0	预留的

15：14	T2C_DELAY	RW	00b	传输端芯片非激活延迟，T2C_DELAY 仅在主模式下使用。它为从设备定义了一个保持时间，该时间在最后一位传输后延迟了多个 SCK 周期的芯片选择去激活。 00： 延迟 1T SCK 01： 延迟 2T SCK 10： 延迟 3T SCK 11： 延迟 4T SCK
13：12	C2T_DELAY	RW	00b	芯片选择-激活-发送-启动延迟。C2T_DELAY 只在主模式下使用。它定义了从设备的设置时间，从芯片选择活动边缘延迟多个 SCK 周期的数据传输。 00： 延迟 1T SCK 01： 延迟 2T SCK 10： 延迟 3T SCK 11： 延迟 4T SCK
11	NSS3_EN	RW	0	NSS3 port 启用位 0： 禁用 1： 启用
10	NSS2_EN	RW	0	NSS2 端口启用位 0： 禁用 1： 启用
9	NSS1_EN	RW	0	NSS1 端口启用位 0： 禁用 1： 启用
8	NSS0_EN	RW	0	NSS0 port 启用位 0： 禁用 1： 启用

7：6	SAMP_PHASE	RW	01b	RX 采样相位，仅在主模式下使用 00： Pre 1T Pclk 周期 01： 未发生 rmal 10： 延迟 1T Pclk 周期 11： 延迟 2T Pclk 周期
5	TXDMA_EN	RW	0	FIFO 模式下启用 TX DMA 0： TX DMA 禁用 1： TX DMA 启用位
4	RXDMA_EN	RW	0	在 FIFO 模式下启用 RX DMA 0： RX DMA 禁用 1： RX DMA 启用位
3：0	CHAR_LEN	RW	0111b	字符长度： 0000，0001，0010： 8-位 0011： 4-位 0100： 5-位 0101： 6-位 0110： 7-位 0111： 8-位 1000： 9-位 1001： 10-位 1010： 11-位 1011： 12-位 1100： 13-位 1101： 14-位 1110： 15-位 1111： 16-位

偏移地址： 17-14h

FIFO 控制寄存器

位	字段名	属性	默认	字段描述
31：14	-	RO	0	预留的
13：9	TX_FIFO_TH	RW	0	发送器 FIFO 触发电平阈值 00000： 0 字符 00001： 1 字符 ..... 10000： 16 字符
8	TX_FIFO_CLR	WO	0	发送器 FIFO 清除 0 = 无作用 1 = 清除发送器 FIFO 并重置发送器 FIFO 计数器
7	-	RO	0	预留的
6：2	RX_FIFO_TH	RW	0	接收器 FIFO 触发电平阈值 00000： 0 字符 00001： 1 字符 ..... 10000： 16 字符
1	RX_FIFO_CLR	WO	0	接收器 FIFO 清除 0 = 无作用 1 = 清除接收器 FIFO 并重置接收器 FIFO 计数器
0	FIFO_EN	RW	0	启用发射机和接收机 FIFOs 模式 0 = 非-FIFO 模式，禁用收发 FIFO，清空 FIFO 指针 1 = FIFO 模式，启用用收发 FIFO

偏移地址： 1B-18h

FIFO 状态寄存器

位	字段名	属性	默认	字段描述
31：21	-	RO	0	预留的
20：16	RX_FIFO_LEN	RO	0	接收器 FIFO 长度指示器位
15：13	-	RO	0	预留的
12：8	TX_FIFO_LEN	RO	0	发送器 FIFO 长度指示器位

7：5	-	RO	0	预留的
4	SPI_BUSY	RO	0	0： SPI 不忙 1： SPI 忙
3	RX_FIFO_FULL	RO	0	接收器 FIFO 全指示器位
2	RX_FIFO_EMPTY	RO	1	接收器 FIFO 空指示位
1	TX_FIFO_FULL	RO	0	发送器 FIFO 全指示器位
0	TX_FIFO_EMPTY	RO	1	发送器 FIFO 空指示位

偏移地址： 1F-1Ch

调试信号寄存器

位	字段名	属性	默认	字段描述
31：24	APB_IF_DBG[7：0]	RO	0	APB 接口模块内部信号
23：12	SPIS_DBG[11：1]	RO	0	SPI 从模块内部信号
12	SPIS_BUSY	RO	0	SPI 从设备忙
11：1	SPIM_DBG[11：1]	RO	0	SPI 主模块内部信号
0	SPIM_BUSY	RO	0	SPI 主设备忙

11 通用异步接收发送器 (UART)

11.1 概述

通用异步接器/发送器(UART)外围设备基于行业标准 TL16C550 异步通信元件，该元件是 TL16C450 的功能升级。功能上类似于 TL16C450 上电(单字符或 TL16C450 模式)，UART 可以置于备用 FIFO (TL16C550)模式。通过对接收和传输的字符进行缓冲，减轻了 CPU 的软件开销。接收器和发送器 FIFO 存储最多 16 个字节，其中包括接收器 FIFO 每个字节的 3 个额外的错误状态位，发送器 FIFO 存储最多 16 个字节，包括接收器 FIFO 每个字节的 3 个额外的错误状态位。

UART 对从外围设备接收到的数据执行串行到并行转换，对从 CPU 接收到的数据执行并行到串行转换。CPU 可以随时读取 UART 状态。该 UART 包括控制能力和一个处理器中断系统，可以量身定制，以最大限度地减少通信链路的软件管理。

该 UART 包括一个可编程的波特率发生器，能够将 UART 输入时钟除以 1 到 65535 的除数，并为内部发送器和接收器逻辑产生 16×参考时钟或 13×参考时钟。 有关 UART 的详细定时和电气规格，请参阅设备特定的数据手册。

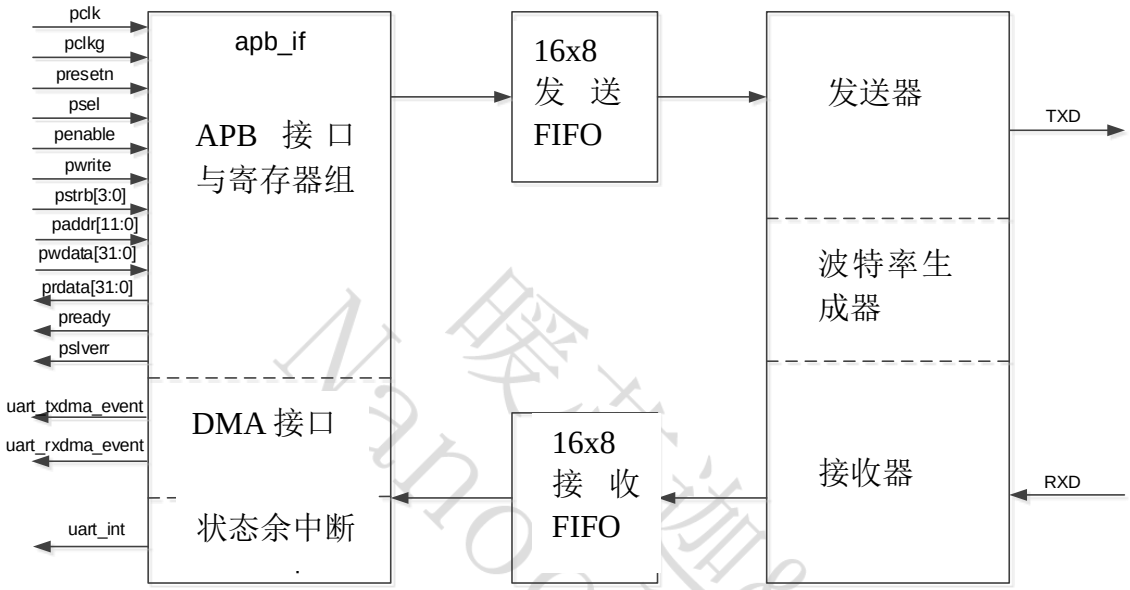
11.1.1 功能列表

- 符合 AMBA APB 规范
- 支持高达 115200bps 的波特率
- 独立地传输和接收 FIFO 缓冲区(16 字节)，以减少中断
- 可编程波特率发生器
- 支持自动流量控制
- 标准异步通信位(启动、停止、奇偶校验)
- DMA 事件支持
- 支持环回测试
- 中断支持
- 全可编程串行接口特点：
  - 数据可以是 5 位、6 位、7 位、8 位
  - 偶数、奇数或无奇偶校验位的生成和检测
  - 1、1.5 或 2 位位生成

11.2 框图

UART 功能框图如下。

图 17 UART 框图



11.3 函数描述

11.3.1 波特率控制

通过配置模式定义寄存器(MDR)中的 OSM_SEL 位来选择 16×或 13×参考时钟。计算除数的公式如下：

- 当 OSM_SEL=0 时，期望波特率=输入时钟频率/((除数+1)× 16)。
- 当 OSM_SEL=1 时，期望波特率=输入时钟频率/((除数+1)× 13)。

UART 在中位周期接收时，16×过采样模式在第 8 个周期采样，13×过采样模式在第 6 个周期采样。

两个 8 位寄存器域(DLH 和 DLL)，称为除数锁存器，用来保存这个 16 位除数。DLH 保存除数的最高有效位，DLL 保存除数的最低有效位。这些除数锁存器必须在 UART 初始化期间加载，以确保波特发生器的预期操作。

11.3.2 协议描述

传输：

UART 发射机部分包括发送器保持寄存器(THR)和发送器移位寄存器(TSR)。当 UART 为 FIFO 模式时，THR 为 16 字节的 FIFO。发送器部分控制是 UART 线路控制寄存器(LCR)的一个函数。 基于 LCR 中选择的设置，UART 发送器将以下内容发送到接收设备：

- 1 起始位
- 5、6、7 或 8 位数据位
- 1 校验位(可选)
- 1、1.5 或 2 个停止位

THR 从内部数据总线接收数据，当 TSR 准备好时，UART 将数据从 THR 移动到 TSR。UART 序列化 TSR 中的数据，并在 uart_txd 引脚上传输数据。在非 FIFO 模式下，如果 THR 为空，并且 THR 空中断处于中断启用寄存器(IER)中，则产生中断。当字符加载到 THR 中时，该中断将被清除。 在 FIFO 模式下，当发送 FIFO 为空时产生中断，当至少有一个字节加载到 FIFO 时中断被清除。

接收：

UART 接收器部分包括接收器移位寄存器(RSR)和接收器缓冲寄存器(RBR)。当 UART 为 FIFO 模式时，RBR 为 16 字节的 FIFO。接收区段控制是 UART 线路控制寄存器(LCR)的一个函数。

基于 LCR 中选择的设置，UART 接收器从发射设备接收以下信息：

- 1 起始位
- 5，6，7，或 8 数据位
- 1 校验位(可选)
- 1 停止位(未检测到与上述数据一起传输的其他停止位)

RSR 从 uart_rxd 引脚接收数据位。然后，RSR 连接数据位，并将结果值移动到 RBR(或接收器 FIFO)。UART 还在每个接收到的字符旁边存储三位错误状态信息，以记录奇偶校验错误、帧错误或中断。

在非 fifo 模式下，当一个字符放置在 RBR 中，并且在中断启用寄存器(IER)中启用接收数据就绪中断时，就会产生一个中断。当从 RBR 中读取字符时，该中断将被清除。在 FIFO 模式下，当 FIFO 被填入到在 FIFO 控制寄存器(FCR)中选择的触发电平时，产生中断，当 FIFO 内容低于触发电平时，中断被清除。

数据格式：

UART 传输格式如下：

1 起始位+数据位(5、6、7、8)+ 1 校验位(可选)+停止位(1、1.5、2)。

根据数据宽度选择它传输 1 个起始位；5、6、7 或 8 个数据位；

如果选择校验位，则 1 校验位；以及 1、1.5 或 2 个停止位，这取决于停止位的选择。

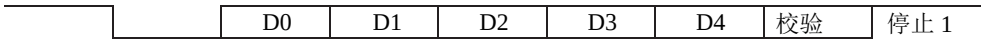
UART 接收格式如下：

1 起始位+数据位(5、6、7、8)+ 1 校验位(可选)+ 1 停止位。

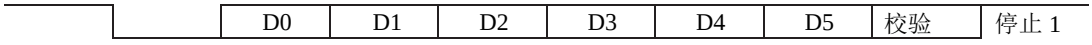
它接收 1 个起始位；5、6、7 或 8 个数据位，这取决于数据宽度选择；如果选择奇偶校验则 1 奇偶校验位，以及 1 个停止位。

协议格式如下所示。

发送/接收 5 位数据，奇偶校验启用，1 停止位



发送/接收 6 位数据，奇偶校验启用，1 停止位



发送/接收 7 位数据，奇偶校验启用，1 停止位

		D0	D1	D2	D3	D4	D5	D6	校验	停止 1

发送/接收 8 位数据，奇偶校验启用，1 停止位

		D0	D1	D2	D3	D4	D5	D6	D7	PARITY	STOP1

11.3.3 FIFO 模式

以下两种模式可用于服务接收器和发送器 FIFOs：

- FIFO 中断模式。FIFO 启用，相关中断禁用。中断发送至 CPU 显示发生特定事件时间。
- FIFO 轮询模式。启用 FIFO，但禁用相关的中断。

CPU 轮询状态位以检测特定事件。

由于接收器 FIFO 和发送器 FIFO 独立控制的，因此可以将其中一个或两者置于中断模式或轮询模式。

FIFO 中断模式：

当在 FIFO 控制寄存器(FCR)中启用接收器 FIFO 和在中断启用寄存器(IER)中启用接收器中断时，为接收器 FIFO 选择中断模式。以下是关于接收器中断的一些要点。

- 当 FIFO 到达在 FCR 中编程的触发电平时，接收器数据就绪中断发送给 CPU。当 CPU 或 DMA 控制器从 FIFO 读取足够的字符使 FIFO 低于它的编程触发电平时它被清除。
- 接收线路状态中断是为了响应超限错误、奇偶错误、帧错误或中断而产生的。此中断比接收端数据就绪中断具有更高的优先级。
- 线路状态寄存器(LSR)中的数据就绪(DR)位表示接收器 FIFO 中字符的存在或缺失。当一个字符从接收移位寄存器(RSR)转移到空的接收器 FIFO 时，DR 位被设置。DR 位保持设置，直到 FIFO 再次为空。
- 如果满足以下所有条件，则发生接收器超时中断。
  - 至少有一个字符在 FIFO 中。
  - 最近的字符是在连续 4 个字符时间之前接收的。字符时间是分配给 1 个起始位、n 个数据位、1 个奇偶校验位和 1 个停止位的时间，其中 n 取决于用线路控制寄存器(LCR)中的 WORD_LEN 位选择的字长。
  - FIFO 的最近一次读取之前已经发生了超过四次连续字符时间。
- 字符时间使用波特率计算。
- 当一个接收器超时中断发生时，当 CPU 或 DMA 控制器从接收器 FIFO 读取一个字符时，它被清除并且超时定时器被清除。如果 FIFO 接收到一个新字符，或者电源管理寄存器中的 URRST 位被清除，中断也会被清除。

- 如果接收器超时中断没有发生，那么在接收到新字符或 CPU 或 DMA 读取接收器 FIFO 之后，超时定时器将被清除。

当在 FCR 中启用发送器 FIFO，并且在 IER 中启用发送器保持寄存器空中断时，将为发送器 FIFO 选择中断模式。当发送器 FIFO 为空时，发送器保持寄存器空中断发生。当发送器保持寄存器(THR)加载(当服务这个中断时，1 到 16 个字符可能被写入发送器 FIFO 时，它被清除。

表 23 UART 字符时间

字长(n)	字符时间	四个字符时间
5	8 位时间	32 位时间
6	9 位时间	36 位时间
7	10 位时间	40 位时间
8	11 位时间	44 位时间

FIFO 轮询模式：

当在 FIFO 控制寄存器(FCR)中启用接收器 FIFO，在中断启用寄存器(IER)中禁用接收器中断时，接收器 FIFO 选择轮询模式。同样，当发送器 FIFO 被启用且发送器中断被禁用时，发送器 FIFO 处于轮询模式。轮询模式下，CPU 通过检查 LSR (线路状态寄存器)中的位数来检测事件：

- RXFIFOE 表示接收器 FIFO 是否存在错误。
- TEMT 位表示发送器保持寄存器(THR)和发送器移位寄存器(TSR)均为空。
- THRE 位表示 THR 为空。
- BI(断点)、FE(分帧错误)、PE(奇偶校验错误)和 OE(超限错误)位指定发生了哪些错误。
- 只要接收器 FIFO 中至少有一个字节，DR(数据就绪)位就会被设置。此外，在 FIFO 轮询模式下：
  - 中断标识寄存器（IIR）不受任何事件的影响，因为中断已被禁用。
  - UART 不指示何时达到接收器 FIFO 触发电平或何时发生接收器超时。

11.3.4 自动硬件流控

通过连接 uart_cts_n 和 uart_rts_n 信号，UART 可以采用自动流控制。uart_cts_n 输入必须在发送器 FIFO 可以传输数据之前激活。当接收器需要更多数据时，uart_rts_n 变为激活状态，并通知发送设备。当 uart_rts_n 连接到 uart_cts_n 时，除非接收器 FIFO 有数据空间，否则不会发生数据传输。因此，启用自动流后，就可以消除超限错误。

RTS 控制流程：

当接收器 FIFO 级别达到触发电平 1、4、8 或 14 时，uart_rts_n 被取消断言。在达到触发电平后，发送 UART 可能会发送一个额外的字节(假设发送 UART 有另一个字节要发送)，因为在开始发送额外的字节之前，它可能不会识别 uart_rts_n 的解除断言。对于 1 级，4 级和 8 级触发电平，当寄存器 RTS_TRI_MODE 为 1 时，当接收器 FIFO 低于触发电平时，uart_rts_n 自动重新生效；当寄存器 RTS_TRI_MODE 为 0 时，接收器 FIFO 为空后，uart_rts_n 自动重新生效。对于触发电平 14，当接收器 FIFO 低于触发电平时，uart_rts_n 会自动重新生效。

CTS 控制流程：

发送器在发送下一个数据字节之前检查 uart_cts_n。如果 uart_cts_n 是激活的，发送器发送下一个字节。为了停止发送器发送以下字节，必须在当前正在发送的最后一个停止位中间之前释放 uart_cts_n。在启用流量控制时，uart_cts_n 级别的变化不会触发中断，因为设备自动控制自己的发送器。如果没有自动流量控制，发送器发送发送器 FIFO 中存在的任何数据，可能会导致接收器超限错误。

11.3.5 回环控制

使用调制解调器控制寄存器(MCR)中的 LOOPBACK_EN 位可以将 UART 置于诊断模式，该位在内部将 UART 输出连接回 UART 输入。在这种模式下，可以在不连接到另一个 UART 的情况下验证发送和接收数据路径、发送器和接收器中断以及调制解调器控制中断。

11.3.6 复位

软件复位

电源管理寄存器控制中的两个位重置 UART 的部分：

- TXRST 位只控制复位发送器。如果 TXRST = 0，发送器是激活的；如果 TXRST = 1，包括 Tx FIFO 在内的发送器处于复位状态。
- RXRST 位只控制接收器的复位。如果 RXRST = 0，接收器是激活的；如果 RXRST = 1，则包括 Rx FIFO 在内的接收器处于复位状态。

在每种情况下，将接收器和/或发送器复位将重置受影响部分的状态机，但不会影响 UART 寄存器。

硬件复位

当处理器复位引脚被断言时，整个处理器被复位并保持在复位状态，直到复位引脚被释放。作为设备复位的一部分，UART 状态机被复位，UART 寄存器被强制恢复到它们的默认状态。

11.3.7 初始化

初始化 UART 需要以下步骤：

1) 将需要的串口硬件引脚设置为复用功能。
2) 通过将适当的时钟除数值写入除数锁存寄存器(DLL 和 DLH)来设置所需的波特率。
3) 如果将使用 FIFO，请选择所需的触发电平，并通过将适当的值写入 FIFO 控制寄存器(FCR)来启用 FIFO。
4) 在配置 FCR 中的其他位之前，必须先设置 FCR 中的 FIFOEN 位。
5) 通过将适当的值写入线路控制寄存器(LCR)来选择所需的协议设置。
6) 如果需要自动流控制，则向调制解调器控制寄存器(MCR)写入适当的值。请注意，并非所有 UART 都支持自动流控制，请参阅特定于设备的数据手册，了解支持的功能。
7) 通过配置 FREE 位选择模拟暂停事件所需的响应，并通过设置电源管理寄存器(PMU)中的 TXRST 和 RXRST 位启用 UART。

11.3.8 中断支持

UART 生成下表中描述的中断请求。所有请求都通过仲裁器多路复用到 CPU 的单个 UART 中断请求。每个中断请求在中断启用寄存器(IER)中都有一个启用，并记录在中断标识寄存器(IIR)中。

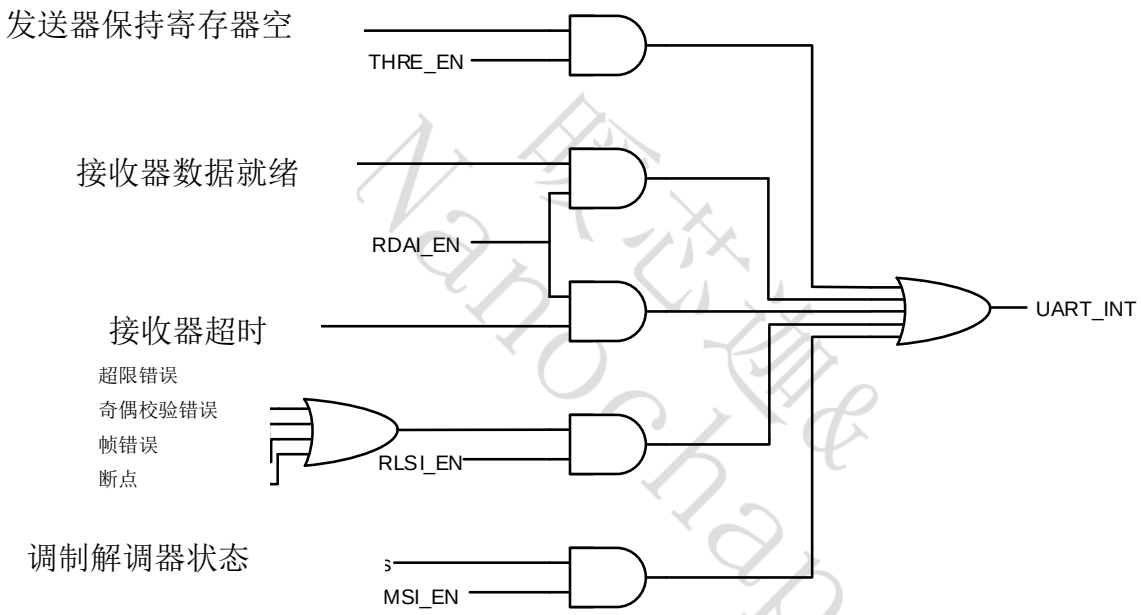
如果发生中断，而对应的启用设置为 1，则中断请求被记录在 IIR 中，并转发到 CPU。如果发生中断，对应的启用被清除为 0，则中断请求被阻塞。中断请求既不记录在 IIR 中也不转发给 CPU。

表 24 UART 中断源

UART 中断请求	中断源	评论
THREINT	THR-空状态：发送器保持寄存器(THR)或发送器 FIFO 为空。所有的数据已经从 THR 复制到发送器移位寄存器(TSR)。	如果在 IER 中启用了 THREINT，通过设置 THRE_EN 位，它将被记录在 IIR 中。 除了使用 THREINT，CPU 还可以轮询线路状态寄存器(LSR)中的 THR 位。
RDAINT	在非 FIFO 模式下接收可用数据或在 FIFO 模式下达到触发电平。	如果在 IER 中启用了 RDAINT，通过设置 RDAI_EN 位，它将被记录在 IIR 中。 除了使用 RDAINT，CPU 可以轮询线路状态寄存器(LSR)中的 DR 位。在 FIFO 模式中，这不是一个功能等效的替代方案，因为 DR 位不响应 FIFO 触发电平。DR 位只表示未读字符的存在或不存在。
RTOINT	接收器超时条件(仅在 FIFO 模式下)： 在最近的 4 个字符时间内，没有字符从接收器 FIFO 中移除或输入，并且在此期间接收器 FIFO 中至少有一个字符。	当接收器 FIFO 电平低于触发电平时，接收器超时中断防止 UART 无限期等待，因此不会产生接收器数据就绪中断。 如果在 IER 中启用了 RTOINT，通过设置 RDAI_EN 位，它将被记录在 IIR 中。 没有状态位来反映超时条件的发生。
RLSINT	接收线路状态条件：发生了超限错误、奇偶错误、帧错误或中断。	• 如果在 IER 中启用了 RLSINT，通过设置 RLSI_EN 位，将 RLSINT 记录在 IIR 中。 • 作为 RLSINT 的替代方案，CPU 可以轮询线路状态寄存器(LSR)中的以下位： 超限错误指示器(OE)、奇偶错误指示器(PE)、分帧错误指示器(FE)和断点指示器(BI)。

MSIINT	调制解调器状态：CTS 更改状态(自动流禁用)，DSR/RI/DCD 更改状态	• 如果在 IER 中启用了 MSIINT，通过设置 MSI_EN 位，将其记录在 IIR 中。 • 作为使用 MSINT 的替代方案，CPU 可以轮询调制解调器状态寄存器(MSR)中的以下位：DELTA_CTS_STS，DELTA_DSR_STS，DELTA_RI_STS 和 DELTA_DCD_STS。

图 18 UART 中断输出



11.4 寄存器

开发者可以访问和控制下面列出的任何 UART 寄存器。这些寄存器控制 UART 操作、接收数据和传输数据，在设备内存映射中的 32 位地址可用。有关这些寄存器的内存地址，请参阅特定于设备的数据手册。

- RBR、THR 和 DLL 共享一个地址，当 LCR 中的 DLAB 位为 0 时，从该地址读取将得到 RBR 的内容，向该地址写入将修改 THR。当 DLAB = 1 时，对该地址的所有访问都将读取或修改 DLL。也可以通过地址偏移 20h 访问 DLL，即专用地址，如果使用专用地址，DLAB 可以为 0，这样始终在共享地址选择 RBR 和 THR。
- IER 和 DLH 共享一个地址，当 DLAB = 0 时，所有访问都读取或修改 IER。当 DLAB = 1 时，所有访问都读取或修改 DLH。 也可以通过地址偏移 24h 访问 DLH，即专用地址，如果使用专用地址，DLAB 可以为 0，这样总是在共享地址处选择 IER。
- IIR 和 FCR 共享一个地址， 无论 DLAB 位的值是多少，从该地址读取将得到 IIR 的内容，而写入将修改 FCR。

表 25 UART 寄存器

偏移	首字母缩写	寄存器描述
00h	RBR	接收器缓冲寄存器(只读取)
00h	THR	发送器保持寄存器(只写入)
04h	IER	中断启用寄存器
08h	IIR	中断识别寄存器(只读取)
08h	FCR	FIFO 控制寄存器 (只写入)
0Ch	LCR	线路控制寄存器
10h	MCR	调制解调器控制寄存器
14h	LSR	线路状态寄存器
18h	MSR	调制解调器状态寄存器
1Ch	SCR	便签寄存器
20h	DLL	除数 LSB 锁存器
24h	DLH	除数 MSB 锁存器
28h	FSR	FIFO 状态寄存器
2Ch	DBG	调试信号寄存器
30h	PMU	电源管理寄存器
34h	MDR	模式定义寄存器

偏移地址： 03-00h

接收缓冲寄存器

位	字段名	属性	默认	字段描述
31：8	-	RO	0	预留的
7：0	RBR	RO	0	UART 接收到的任何数据字节都可以通过读取这个寄存器来访问

偏移地址： 03-00h

发送器保持寄存器

位	字段名	属性	默认	字段描述
31：8	-	RO	0	预留的
7：0	THR	WO	0	这个寄存器是用来缓冲输出字符的

偏移地址： 07-04h

中断启用寄存器

位	字段名	属性	默认	字段描述
31：4	-	RO	0	预留的
3	MSI_EN	RW	0	启用调制解调器状态中断
2	RLSI_EN	RW	0	启用接收线路状态中断 0 = 启用接收线路状态中断 1 = 启用接收线路状态中断
1	THRE_EN	RW	0	启用发送器保持寄存器空中断 0 =禁用发送器保持寄存器空中断 1 = 启用发送器保持寄存器空中断
0	RDAI_EN	RW	0	启用接收器数据可用中断和字符超时指示中断 0 = 禁用接收数据可用中断和字符超时指示中断 1 = 启用接收数据可用中断和字符超时指示中断

偏移地址： 0B-08h

中断识别寄存器

位	字段名	属性	默认	字段描述
31：8	-	RO	0	预留的
7：6	FIFO_USE	RO	00b	FIFO 使用指示器 0 = 非-FIFO 模式 1 = 预留的 2 = 预留的 3 = 启用 FIFO，FIFO 控制寄存器(FCR)中的 FIFOEN 位设置为 1
5：4	-	RO	0	预留的
3：1	INT_类型	RO	000b	中断类型 0 = 调制解调器状态 1 = 发送器保持寄存器空(优先级 3) 2 = 接收器数据可用(优先级 2) 3 = 接收线路状态(优先级 1，最高) 4 = 预留的 5 = 预留的 6 = 字符超时指示(优先级 2) 7 = 预留的
0	INT_PEND	RO	1	中断挂起 当任何 UART 中断生成并在 IER 中启用时，IPEND 强制为 0。IPEND 保持为 0，直到所有挂起的中断都清除或硬件复位。如果没有启用中断，则 IPEND 永远不会强制为 0。 0 = 中断挂起 1 = 中断未挂起

中断识别和中断清除信息

优先级	3	2	1	0	中断类型	中断源	清除中断的事件
无	0	0	0	1	无	无	无
1	0	1	1	0	接收器线路状态	检测到超限错误、奇偶错误、分帧错误或断点。	对于超限错误，读取线路状态寄存器(LSR)可以清除中断。 对于奇偶校验错误、帧错误或断点，只有在所有错误数据被读取后，中断才会被清除。
2	0	1	0	0	接收器数据就绪	非 FIFO 模式：接收器数据已准备就绪。	非-FIFO 模式：读取接收器缓冲寄存器(RBR)。
						FIFO 模式：已达到触发电平。如果经过 4 个字符时间，但没有访问 FIFO，则再次断言中断。	FIFO 模式：FIFO 下降到触发电平以下。
2	1	1	0	0	接收器超时	仅 FIFO 模式：在过去的 4 个字符时间里，没有字符从接收器 FIFO 中移除或输入，并且在这段时间内，接收器 FIFO 中至少有一个字符。	下列事件之一： • 从接收器 FIFO 读取一个字符。 • 一个新字符到达接收器 FIFO。 • 电源管理寄存器(PMU)的 URRST 位被加载为 1。
3	0	0	1	0	发送器保持寄存器空	• 非-FIFO 模式：发送器保持寄存器(THR)为空。 • FIFO 模式：发送器 FIFO 为空。	一个字符被写入发送器保持寄存器(THR)。

偏移地址： 0B-08h

FIFO 控制寄存器

位	字段名	属性	默认	字段描述
31：8	-	RO	0	预留的
7：6	RXFIFOTL	WO	00b	接收器 FIFO 触发电平。RXFIFTL 设置接收器 FIFO 的触发电平。当达到触发电平,生成接收器数据就绪中断(如果启用了中断请求)。当 FIFO 低于触发电平时，中断被清除。 0 = 1 字节 1 = 4 字节 2 = 8 字节 3 = 14 字节

5：4	-	RO	0	预留的
3	DMA_MODE	WO	0	如果 FIFO 被启用，DMA MODE1 将被启用，始终将 1 写入 DMAMODE1，硬件复位后，将 DMAMODE1 从 0 修改为 1，DMAMOD1 = 1 是 UART 和 EDMA 控制器之间正确通信的必要条件。 0 = 禁用 DMA MODE1 1 = 启用 DMA MODE1
2	TXCLR	WO	0	DMAMOD1 = 1 是 UART 和 EDMA 控制器之间正确通信的必要条件。 0 = 无作用 1 = 清除发送器 FIFO，复位发送器 FIFO 计数器 移位寄存器未被清除
1	RXCLR	WO	0	接收器 FIFO 清除，写一个 1 到 RXCLR 以清除该位。 0 = 无作用 1 = 清除接收器 FIFO 并复位接收器 FIFO 计数器。移位寄存器未被清除
0	FIFOEN	WO	0	启用发送器和接收器 FIFO 模式，FIFOEN 必须在写入其他 FCR 位或未编程 FCR 位之前复位，清除此位将清除 FIFO 计数器。 0 = 非-FIFO 模式，禁用收发 FIFO，清空 FIFO 指针 1 = FIFO 模式，启用发送器和接收器的 FIFO

偏移地址： 0F-0Ch

线路控制寄存器

位	字段名	属性	默认	字段描述
31：8	-	RO	0	预留的
7	DLAB	RW	0	除数锁存器访问位，可以在专用地址或由 RBR、THR 和 IER 共享的地址访问除数锁存器寄存器(DLL 和 DLH)。使用共享地址需要切换 DLAB 来更改所选择的寄存器。如果使用专用地址，DLAB 可以= 0。 0 = 允许访问接收器缓冲寄存器(RBR)，发送器保持寄存器(THR)和中断启用寄存器(IER)。 在 RBR、THR 和 DLL 共享的地址上，CPU 可以从 RBR 读取数据，也可以向 THR 写入数据。 在 IER 和 DLH 共享的地址上，CPU 可以对 IER 进行读写。 1 = 允许在读或写操作(DLL 和 DLH)期间访问波特发生器的除数锁存器，在 RBR、THR 和 DLL 共享的地址上，CPU 可以对 DLL 进行读写。在 IER 和 DLH 共享的地址上，CPU 可以对 DLH 进行读写。

6	BREAK_EN	RW	0	断点控制。 0 = 已禁用断点条件 1 = 断点状态被传送到接收 UART，在断点条件中，uart_txd 信号被强制设置为间隔(清除)状态
5	STICK_EN	RW	0	固定奇偶校验启用，STICK_EN 位与 EVEN_EN 和 PARITY_EN 位协同工作， STICK_EN、EVEN_EN 和 PARITY_EN 三位之间的关系如下表所示。 0 = 禁用固定奇偶校验 1 = 启用固定奇偶校验 • 当选择奇偶校验(EVEN_EN = 0)时，传输奇偶校验位，并按设置校验。 •当选择偶校验(EVEN_EN = 1)时，传输校验位，校验为清除。
4	EVEN_EN	RW	0	偶校验选择，启用奇偶校验时选择奇偶校验(PARITY_EN = 1)，EVEN_EN 位与 STICK_EN 和 PARITY_EN 位协同工作，STICK_EN、EVEN_EN 和 PARITY_EN 三位之间的关系如下表所示。 0 = 选择奇数奇偶校验(在数据位和奇偶校验位中传输或检查奇数个逻辑 1) 1 = 选择偶数奇偶校验(在数据和奇偶校验位中传输或校验偶数个逻辑 1)
3	PARITY_EN	RW	0	启用奇偶校验，PARITY_EN 位与 EVEN_EN 和 STICK_EN 位协同工作，SP 位、EPS 位和 PEN 位之间的关系如下表所示。 0 = 没有传输或检查奇偶校验位 1 = 在传输数据中产生奇偶校验位，并在接收数据中在最后一个数据字位和第一个停止位之间进行校验
2	STOP_LEN	RW	0	产生的停止位数，STOP_LEN 指定了每个要传输的字符中的 1、1.5 或 2 个停止位，当 STOP_LEN = 1 时，WORD_LEN 位决定停止位的个数， 不管选择了多少个停止位，接收器只计时第一个停止位。 表 3-10 汇总了停止位产生的个数。 0 = 产生一个停止位 1 = STOP_LEN bit 确定停止位的个数： »当 STOP_LEN = 0 时，将产生 1.5 个停止位。 » 当 STOP_LEN = 1h、2h 或 3h 时，将产生 2 个停止位。

1：0	WORD_LEN	RW	11b	字长选择，每个发送或接收的串行字符的位数。当 STOP_LEN = 1 时，WORD_LEN 位决定停止位的个数。 0 = 5 位 1 = 6 位 2 = 7 位 3 = 8 位

STICK_EN	EVEN_EN	PARITY_EN	奇偶校验选项
x	x	0	禁用奇偶校验： 没有传输或检查奇偶校验位。
0	0	1	奇偶校验选择：逻辑 1s 的奇数个数。
0	1	1	偶校验选择：逻辑 1s 的偶数个数。
1	0	1	选择奇偶校验，传输奇偶校验位，并按设定进行校验。
1	1	1	选择奇偶校验位传输奇偶校验，并检查为清除。

偏移地址： 13-10h

调制解调器控制寄存器

位	字段名	属性	默认	字段描述
31：7	-	RO	0	预留的
6	RTS_TRI_MODE	RW	0	RTS 触发模式 0： 对于 1、4 和 8 触发电平,uart_rts_n 在接收器 FIFO 为空后自动重新生效 1： 对于 1、4 和 8 触发电平，当接收器 FIFO 低于触发电平时，uart_rts_n 会自动重新生效
5	AUTOFLOW_EN	RW	0	自动流量控制启用
4	LOOPBACK_EN	RW	0	开启环回模式，LOOPBACK_EN 用于环回功能的诊断测试。 0 = 禁用环回模式 1 = 启用用环回模式，当设置 LOOPBACK_EN 时，会发生以下情况： » uart_txd 信号被设置为高 » uart_rxd 引脚断开 »发送器移位寄存器(TSR)的输出被循环回接收器移位寄存器(RSR)的输入

3	AUX2	RW	0	AUX2 控制位，当处于环回模式时，它连接到 DCD
2	AUX1	RW	0	AUX1 控制位，在环回模式下，它连接到 RI
1	RTS_CTRL	RW	0	当自动流量控制被禁用时，RTS 由软件控制 1： RTS 为“0” 0： RTS 为“1”
0	DTR_CTRL	RW	0	DTR 控制，在环回模式下，它连接到 DSR 1： DTR 为 “0” 0： DTR 为 “1”

偏移地址： 17-14h

线路状态寄存器

位	字段名	属性	默认	字段描述
31：8	-	RO	0	预留的
7	RECEIVER_ERROR	RO	0	接收器 FIFO 错误 处于非 FIFO 模式： 0 = 没有错误，或者 RXFIFOE 被清除，因为 CPU 从接收缓冲区寄存器(RBR)中读取了错误的字符 1 = 接收缓冲寄存器(RBR)中有奇偶校验错误、帧错误或断点指示器 在 FIFO 模式： 0 = 没有错误，或者 RXFIFOE 被清除，因为 CPU 从接收器 FIFO 读取了错误字符，并且接收器 FIFO 没有错误 1 = 接收器 FIFO 中至少有一个奇偶校验错误、帧错误或断点指示器
6	TRANSMITTER_EMPTY	RO	1	发送器空(TEMT)指示器 在非 FIFO 模式： 0 = 发送器保持寄存器(THR)或发送器移位寄存器(TSR)都包含一个数据字符 1 = 发送器保持寄存器(THR)和发送器移位寄存器(TSR)均为空 在 FIFO 模式： 0 = 发送器 FIFO 或发送器移位寄存器(TSR)都包含一个数据字符 1 = 发送器 FIFO 和发送器移位寄存器(TSR)均为空

5	THR_EMPTY	RO	1	发送器保持寄存器空(THRE)指示器，如果设置了 THRE 位，并设置了相应的中断启用(IER 中的 THRE_EN = 1)，则会生成中断请求。 在非 FIFO 模式： 0 = 发送器保持寄存器(THR)不为空，THR 已被 CPU 加载。 1 = 发送器保持寄存器(THR)为空(准备接受一个新字符)，THR 的内容已传输到发送器移位寄存器(TSR)。 在 FIFO 模式： 0 = 发送器 FIFO 非空，至少有一个字符被写入发送器 FIFO。如果发送器 FIFO 未满，则可以写入该 FIFO。 1 =发送器 FIFO 为空。FIFO 中的最后一个字符已被传输到发送器移位寄存器(TSR)。
4	BREAK_ERROR_STS	RO	0	断点指示器，当接收数据输入保持低电平的时间超过一个整字传输时间时，设置 BI 位。整字传输时间被定义为传输起始、数据、校验和停止位的总时间。如果设置了 BI 位，并且设置了相应的中断启用(IER 中的 RLSI_EN = 1)，则会产生中断请求。 在非 FIFO 模式： 0 = 没有检测到断点，或者 BI 位被清除，因为 CPU 从接收缓冲区寄存器(RBR)读取了错误的字符。 1 = 在接收缓冲区寄存器(RBR)中检测到字符的断点。 In FIFO 模式： 0 = 没有检测到断行，或者 BI 位被清除，因为 CPU 从接收器 FIFO 读取了错误字符，而下一个要从 FIFO 读取的字符没有断行指示。 1 = 在接收器 FIFO 顶部检测到字符断点。
3	FRAME_ERROR_STS	RO	0	帧误差(FE)指示器，当接收到的字符没有有效的停止位时，会发生帧错误，为了响应帧错误，UART 设置 FE 位并等待直到 RX 引脚上的信号变高，当 RX 信号变高时，接收器准备检测新的起始位并接收新的数据。如果设置 FE 位，并设置相应的中断启用(IER 中的 RLSI_EN = 1)，则产生一个中断请求。 在非 FIFO 模式： 0 = 因为 CPU 从接收缓冲区寄存器(RBR)读取了错误数据，所以没有检测到帧错误，或者 FE 位被清除。 1 = 在接收缓冲区寄存器(RBR)中检测到字符的帧错误。 在 FIFO 模式： 0 = 由于 CPU 从接收器 FIFO 读取错误数据，且下一个要从 FIFO 读取的字符没有帧错误，所以没有检测到帧错误，或者 FE 位被清除。 1 =在接收器 FIFO 顶部的字符检测到帧错误。

2	PARITY_ERROR_STS	RO	0	奇偶校验错误（PE）指示器。当接收字符的奇偶校验与线路控制寄存器(LCR)中的 EVEN_EN 位选择的奇偶校验不匹配时，就会发生奇偶校验错误。如果设置 PE 位，并设置相应的中断启用(IER 中的 RLSI_EN = 1)，则产生一个中断请求。 在非 FIFO 模式： 0 = 因为 CPU 从接收缓冲区寄存器(RBR)读取了错误的数据，所以没有检测到校验错误，或者 PE 位被清除。 1 = 接收缓冲寄存器(RBR)中的字符被检测到奇偶校验错误。 在 FIFO 模式： 0 = 因为 CPU 从接收 FIFO 读取错误数据，并且下一个要从 FIFO 读取的字符没有校验错误，所以没有检测到校验错误，或者 PE 位被清除。 1 = 在接收器 FIFO 顶部的字符检测到奇偶校验错误。
1	OVERRUN_ERROR_STS	RO	0	超限 (OE)指示器，非 FIFO 模式下的超限错误和 FIFO 模式下的超限错误是不同的，如果设置了 OE 位，并且设置了相应的中断启用(IER 中的 RLSI_EN = 1)，则会产生中断请求。 在非 FIFO 模式： 0 = 因为 CPU 读取了线路状态寄存器(LSR)的内容，所以没有检测到超限错误，或者已清除 OE 位。 1 =检测到超限错误，在接收缓冲区寄存器(RBR)中的字符被读取之前，它会被 RBR 中的下一个字符覆盖。 在 FIFO 模式： 0 = 因为 CPU 读取了线路状态寄存器(LSR)的内容，所以未检测到超限错误，或已清除 OE 位。 1 = 检测到超限错误， 如果数据继续填满 FIFO 超过触发水平，只有在 FIFO 已满且移位寄存器中已完全接收到下一个字符后，才会发生超限错误。一旦发生超限错误，就会向 CPU 显示。 新字符覆盖移位寄存器中的字符，但不会转移到 FIFO。

0	DATA_READY	RO	0	接收器的数据就绪(DR)指示器。如果设置了 DR 位，并且设置了相应的中断启用 (在 IER 中 RDAI_EN = 1)，则产生中断请求。 在非 FIFO 模式： 0 = 因为字符是从接收器缓冲区寄存器(RBR)读取的，所以数据未就绪或 DR 位被清除。 1 = 数据准备就绪，一个完整的输入字符已经被接收并传输到接收器缓冲寄存器(RBR)。 在 FIFO 模式： 0 = 因为已读取接收器 FIFO 中的所有字符都，所以数据未就绪或 DR 位被清除。 1 = 数据准备就绪，接收器 FIFO 中至少有一个未读字符。如果 FIFO 为空，则在接收到完整的输入字符并将其传输到 FIFO 时立即设置 DR 位。DR 位保持设置，直到 FIFO 再次为空。

偏移地址： 1B-18h

调制解调器状态寄存器

位	字段名	属性	默认	字段描述
31：8	-	RO	0	预留的
7	MSR_DCD	RO	0	载波检测输入的补码，当 UART 处于诊断测试模式(环回模式 MCR[4] = 1)时，该位等于 MCR 位 3 (AUX2)。
6	MSR_RI	RO	0	环形指示器输入的补码，当 UART 处于诊断测试模式(环回模式 MCR[4] = 1)时，该位等于 MCR 位 2 (AUX1)。
5	MSR_DSR	RO	0	数据集就绪输入的补码，当 UART 处于诊断测试模式(环回模式 MCR[4] = 1)时，该位等于 MCR 位 0 (DTR)。
4	MSR_CTS	RO	0	“清除发送”输入的补码，当 UART 处于诊断测试模式(环回模式 MCR[4] = 1)时，该位等于 MCR 位 1 (RTS)。
3	DELTA_DCD_STS	RO	0	DCD 指示器位变化，DCD 表示自上次 CPU 读取 DCD 输入以来，它已经改变了状态。当设置了 DCD 并启用了调制解调器状态中断时，将生成一个调制解调器状态中断。
2	DELTA_RI_STS	RO	0	RI (TERI)指示位的后沿，TERI 表示 RI 输入由低变为高。当设置了 TERI 并且启用了调制解调器状态中断时，将生成一个调制解调器状态中断。
1	DELTA_DSR_STS	RO	0	DSR 指示器位的变化，DDSR 指示 DSR 输入自从上次被 CPU 读取以来已经改变了状态。当设置了 DDSR 并启用了调制解调器状态中断时，将生成一个调制解调器状态中断。

0	DELTA_CTS_STS	RO	0	CTS 指示器位变化，DCTS 表示 CTS 输入自从上次被 CPU 读取以来已经改变了状态，当设置 DCTS(未启用自动流控制且启用调制解调器状态中断)时，将生成一个调制解调器状态中断。当启用自动流控制时，不会产生中断。

偏移地址： 1F-1Ch

便签寄存器

位	字段名	属性	默认	字段描述
31：8	-	RO	0	预留的
7：0	SCR	RW	0	这些位用于程序员作为便签，因为它暂时保存了程序员的数据，而不影响任何其他 UART 操作。

偏移地址： 23-20h

除数 LSB 锁存器

位	字段名	属性	默认	字段描述
31：8	-	RO	0	预留的
7：0	DLL	RW	0	16 位除数的 8 个最低有效位(LSB) 当 OSM_SEL=0 时，期望波特率=输入时钟频率 /(({DLH，DLL}+1) × 16) 当 OSM_SEL=1 时，期望波特率=输入时钟频率 /(({DLH，DLL}+1) × 13)

偏移地址： 27-24h

除数 MSB 锁存器

位	字段名	属性	默认	字段描述
31：8	-	RO	0	预留的
7：0	DLH	RW	0	16 位除数的 8 个最高有效位(MSB) 当 OSM_SEL=0 时，期望波特率=输入时钟频率 /(({DLH，DLL}+1) × 16) 当 OSM_SEL=1 时，期望波特率=输入时钟频率 /(({DLH，DLL}+1) × 13)

偏移地址： 2B-28h

FIFO 状态寄存器

位	字段名	属性	默认	字段描述
31：24	-	RO	0	预留的
23：21	-	RO	0	预留的
20：16	RX_FIFO_LEN	RO	0	接收器 FIFO 长度指示位
15：13	-	RO	0	预留的
12：8	TX_FIFO_LEN	RO	0	发送器 FIFO 长度指示位
7：4	-	RO	0	预留的
3	RX_FIFO_FULL	RO	0	接收器 FIFO 全指示器位
2	RX_FIFO_EMPTY	RO	1	接收器 FIFO 空指示位
1	TX_FIFO_FULL	RO	0	发送器 FIFO 全指示器位
0	TX_FIFO_EMPTY	RO	1	发送器 FIFO 空指示位

偏移地址： 2F-2Ch

调试信号寄存器

位	字段名	属性	默认	字段描述
31：13	-	RO	0	预留的
12	RX_FIFO_TIMEOUT	RO	0	表示没有访问接收器 FIFO，这会导致超时
11	RX_BUF_FULL	RO	0	接收器缓冲区全指示位
10	TX_BUF_EMPTY	RO	1	发送器缓冲区空指示位
9	RX_BAUDRATE	RO	0	接收器波特率
8	TX_BAUDRATE	RO	0	发送器波特率

7：5	RX_CUR_STATE	RO	0	接收器 FSM
4	RXD_WORK	RO	0	表示接收器在工作
3：1	TX_CUR_STATE	RO	0	发送器 FSM
0	TXD_WORK	RO	0	表示发送器在工作

偏移地址： 33-30h

电源管理寄存器

位	字段名	属性	默认	字段描述
31：2	-	RO	0	预留的
1	TXRST	RW	0	UART 发送器复位，复位并启用发送器 1 =发送器被禁用并处于复位状态 0 = 启用发送器
0	RXRST	RW	0	UART 接收器复位，复位并启用接收器 1 = 接收器被禁用并处于复位状态 0 = 启用接收器

偏移地址： 37-34h

模式定义寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	OSM_SEL	RW	0	过采样模式选择 0 = 16× 过采样 1 = 13× 过采样

12 集成电路间接口(I2C)

12.1 概述

I2C (内部集成电路)总线接口是单片机与I2C串行总线之间的接口。它提供多主功能，控制所有I2C总线的时序、协议、仲裁和定时。它支持标准模式(Sm，最高100 kHz)和Fm模式(Fm，最高400 kHz)和HS模式。

根据特定的设备实现，DMA能力可以用于减少CPU过载。

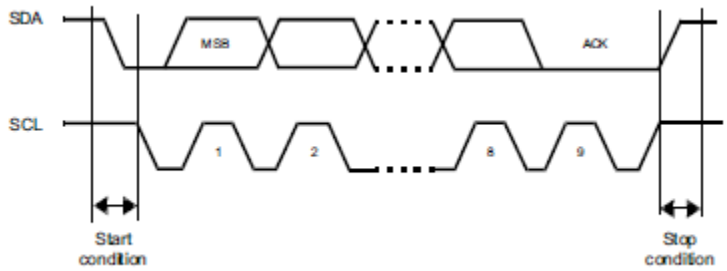
在主模式下，I2C接口发起数据传输并产生时钟信号。串行数据传输总是以启动条件开始，以停止条件结束。启动和停止条件都是由软件在主模式下生成的。

在从模式下，接口能够识别自己的地址(7位或10位)和通用呼叫地址。通用呼叫地址检测可以由软件启用或禁用。

数据和地址以8位字节传输，首先是MSB。起始条件之后的第一个字节包含地址(一个是7位模式，两个是10位模式)。该地址总是以主模式传输。

第9个时钟脉冲跟随字节传输的8个时钟周期，在此期间，接收器必须向发送器发送一个确认位。

图 19 I2C 总线协议



可以由软件启用或禁用确认。I2C接口地址(7位/ 10位寻址或通用呼叫地址)可由软件选择。

12.1.1 功能列表

- 符合 AMBA APB 规范
- 支持飞利浦 I2C 标准
- 同一接口可以作为主接口或从接口
- 半双工通信(只有发射器或接收器)
- 7 位/10 位寻址的生成和检测
- 支持不同的通信速度：

– 标准速度(高达 100khz)

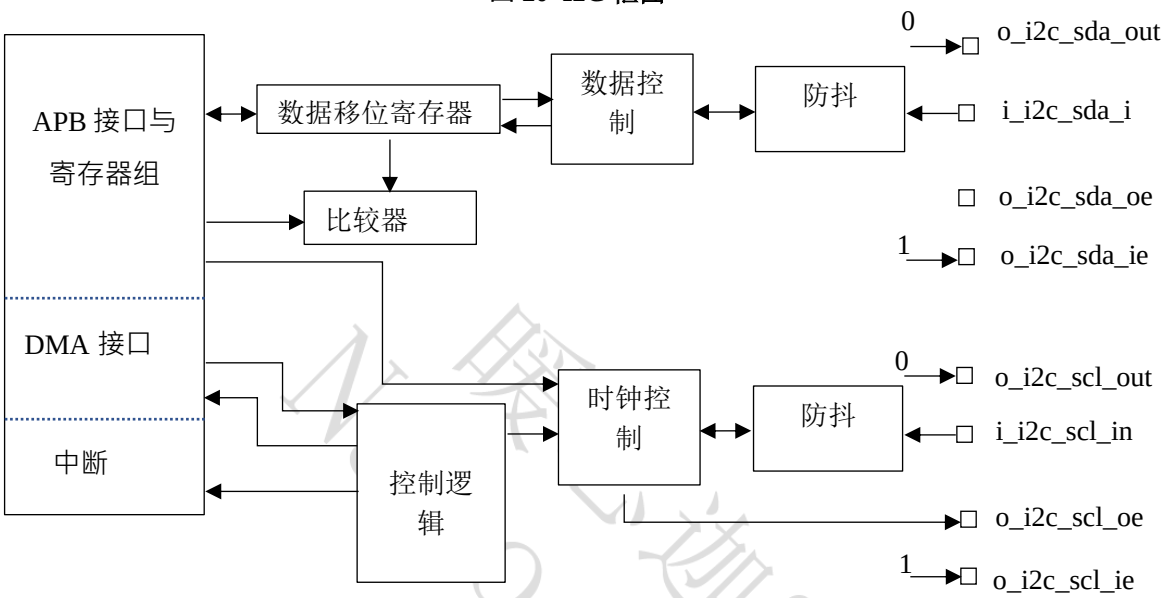
– 快速度(高达 400khz)

- 模拟噪声滤波器
- 状态标志：
  - 发送器/接收器模式标志
  - 字节结束传输标志
  - I2C 忙碌标志
- 数据方向总是 MSB 优先
- 错误标志：
  - 主模式仲裁丢失条件
  - 地址/数据传输后确认失败
  - 检测错误的启动或停止条件
- 具有 DMA 功能的 1 字节缓冲区

12.2 框图

I2C 接口的框图如下。

图 20 I2C 框图



12.3 函数描述

12.3.1 支持模式

接口有以下四种工作方式：

- 从发送器
- 从接收器
- 主发送器
- 主接收器

12.3.2 I2C 从模式

默认情况下,I2C 接口以从模式运行。要从默认从模式切换到主模式，需要生成起始条件。

外设输入时钟必须在 I2C_CR2 寄存器中编程，以生成正确的计时时序。

从发送器

1) 配置 I2C_CR2 寄存器中断；
2) 配置 I2C_CR1 的 PE 启用 I2C 接口；
3) 配置 I2C_OAR 设置 I2C 从地址；
4) 配置 I2C_CR1 的 ACK 设置 ack 位；
5) 读取 I2C_SR 的 ADDR，清除 ADDR 位；
6) 写入 I2C_DR 寄存器；
7) 读取 I2C_SR 的 AF 和 STOPF；
8) 写入 I2C_CR1 的 PE 禁用 I2C 接口。

如果设置了 TxE，并且在下一次数据传输结束之前没有将一些数据写入 I2C_DR 寄存器，设置 BTF 位，接口一直等待，直到读取 I2C_SR，然后写入 I2C_DR 寄存器，清除 BTF，将 SCL 拉低。

从接收器

1) 配置 I2C_CR2 寄存器中断；
2) 配置 I2C_CR1 的 PE 启用 I2C 接口；
3) 配置 I2C_OAR 设置 I2C 从地址；
4) 配置 I2C_CR1 的 ACK 设置 ack 位；

5) 读取 I2C_SR 的 ADDR，清除 addr 位；

6) 设置 I2C_SR 的 RXNE，读取 I2C_DR 清除 RXNE 位；

7) 读取 I2C_SR 的 STOPF；

8) 写入 I2C_CR1 的 PE 禁用 I2C 接口。

如果设置了 RxNE，并且在下一次数据接收结束前不读取 DR 寄存器中的数据，设置 BTF 位，接口一直等待，直到从 I2C_DR 寄存器读取 BTF 清除，将 SCL 拉低。

12.3.3 I2C 主模式

在主模式下，I2C 接口发起数据传输并产生时钟信号。串行数据传输总是以起始条件开始，以停止条件结束。只要总线上有一个启动位产生启动条件，就会选择主模式。

主发送器

1) 配置 I2C_CR2 寄存器的中断和时钟频率；

2) 配置 I2C_CR1 的 PE 启用 I2C 接口；

3) 配置 I2C_CR1 的 START 启动 I2C 主设备；

4) 读取 I2C_SR 的 SB；

5) 用地址写入 I2C_DR 寄存器清除 SB，在 7 位主发送器写从地址，在 10 位主发送器写 10 位地址报头；

6) 在 7 位主发送器读取 I2C_SR 的 ADDR 清除，10 位主发送器读取 I2C_SR 的 ADDR10 并写入 I2C_DR 的从地址清除 ADDR10；

7) 在 7 位主发送器用数据写入 I2C_DR，在 10 位主发送器读取 I2C_SR 的 ADDR，清除并写入 I2C_DR；

8) 读取 I2C_SR 的 BTF；

9) 写入 I2C_CR1 以设置停止位。

如果设置了 TxE，并且在最后一次数据传输结束前没有将数据字节写入 DR 寄存器，设置 BTF，接口等待直到写入 I2C_DR 清除 BTF，将 SCL 拉低。

主接收器

1) 配置 I2C_CR2 寄存器的中断和时钟频率；

2) 配置 I2C_CR1 的 PE 启用 I2C 接口；

3) 配置 I2C_CR1 的 START 启动 I2C 主设备；

4) 读取 I2C_SR 的 SB；

5) 用地址写 I2C_DR 寄存器清除 SB，在 7 位主写从地址，在 10 位主写 10 位地址报头；

6) 7 位主读取 I2C_SR 的 ADDR 清除，10 位主读取 I2C_SR 的 ADDR10 并用从地址写入 I2C_DR 清除 ADDR10，写入 I2C_CR1 设置 START。

7) 在 7 位主位中读取 I2C_SR 的 RXNE 并读取 2C_DR，在 10 位主位中读取 I2C_SR 的 ADDR 以清除并写入 I2C_CR1 以设置起始位；

8) 在 7 位的主设备中，如果接收到最后的数据，则通过读取 I2C_DR 寄存器清除 RxNE=1，写入 ACK=0 并停止 I2C_CR1；在 10 位主设备中读取 I2C_SR 的 SB，并写入带有 10 位地址头的 I2C_DR 寄存器以清除 SB；

9) 在 10 位主设备中读取 I2C_SR 的 ADDR 以清除；

10) 在 10 位主设备中读取 I2C_SR 的 ADDR 以清除；

11) 在 10-位主设备中，如果接收到最后的数据，通过读取 I2C_DR 寄存器清除 RxNE=1，并写入 I2C_CR1 的 ACK=0 和 STOP；

12) 读取 I2C_DR 寄存器清除 RxNE=1。

12.3.4 错误条件

以下是可能导致通信失败的错误条件。

1) 总线错误(BERR)

2) 确认失败(AF)

3) 仲裁丢失(ARLO)

4) 超限/欠载错误(OVR)

12.3.5 I2C 中断

表 26 中断表

中断事件	事件标志	启用控制位
发送起始位(主设备)	SB	ITEVFEN
发送的地址(主设备)或匹配的地址(从设备)	ADDR	
发送的 10 位报头(主设备)	ADD10	
接收停止(从设备)	STOPF	
数据字节传输完成	BTF	
接收缓冲区非空	RxNE	ITEVFEN 和 ITBUFEN
发送缓冲区空	TxE	
总线错误	BERR	ITERREN
仲裁丢失(主设备)	ARLO	
确认失败	AF	
超限/欠载	OVR	

12.4 寄存器

必须通过字节(8 位)、半字(16 位)或字(32 位)访问外设寄存器。这些寄存器在设备内存映射中的 32 位地址可用。有关这些寄存器的内存地址，请参阅特定于设备的数据手册。

表 27 I2C 寄存器

偏移	首字母缩写	寄存器描述
00h	I2C_DR	I2C 数据寄存器
04h	I2C_OAR	I2C 自有地址寄存器
08h	I2C_CR1	I2C 控制寄存器 1
0Ch	I2C_CR2	I2C 控制寄存器 2
10h	I2C_SR	I2C 状态寄存器
14h	I2C_DBG	I2C 调试数据寄存器

偏移地址： 03-00h

I2C 数据寄存器

位	字段名	属性	默认	字段描述
31：8	-	RO	0	预留的
7：0	DATA	RW	0	接收到的或要传输到总线的字节

偏移地址： 07-04h

I2C 自有地址寄存器

位	字段名	属性	默认	字段描述
31：10	-	-	0	预留的
9：8	ADD[9：8]	RW	0	7-位寻址模式： 忽略 10-位寻址模式： 地址的第 9：8 位
7：1	ADD[7：1]	RW	0	地址的第 7：1 位

0	ADD0	RW	0	7-位寻址模式： 忽略 10-位寻址模式： 地址第 0 位

偏移地址： 0B-08h

I2C 控制寄存器 1

位	字段名	属性	默认	字段描述
31：16	-	-	0	预留的
15	SWRST	RW	0	软件复位
14：8	-	-	0	预留的
7	DBYPASS	RW	0	防抖屏蔽 0： 禁用屏蔽 1： 启用屏蔽
6	MMASTER	RW	0	多主设备启用 0： 禁用多主设备 1： 启用多主设备
5	ACK	RW	0	确认启用 0： 未返回确认 1： 收到字节后返回的确认(匹配的地址或数据)
4	STOP	RW		停止生成 在主模式： 0： 未停止生成 1： 在当前字节传输或发送当前启动条件后停止生成。 在从模式： 0： 未停止生成 1： 在当前字节传输后释放 SCL 和 SDA 行

3	START	RW	0	起始生成 在主模式： 0： 无起始生成 1： 重复起始生成 在从模式： 0： 无起始生成 1： 在总线空闲时起始生成
2	NOSTRETCH	RW	0	时钟拉伸禁用(从模式) 0：启用时钟拉伸 1：禁用时钟拉伸
1	ENGC	RW	0	通用呼叫启用 0： 禁用通用呼叫，地址 00h 为 NACKed. 1：启用通用呼叫，地址 00h 为 ACKed
0	PE	RW	0	外设启用 0： 禁用外设 1： 启用外设

偏移地址： 0F-0Ch

I2C 控制寄存器 2

位	字段名	属性	默认	字段描述
31：11	-	-	0	预留的
10	LAST	RW	0	DMA 最后传输 0： 下一次 DMA EOT 不是最后一次传输 1： 下一次 DMA EOT 是最后一次传输
9	DMAEN	RW	0	DMA 请求启用 0： DMA 请求禁用 1： 当 TxE=1 或 RxNE =1 时 DMA 请求启用

8	ITBUFEN	RW	0	启用缓冲区中断 0： TxE = 1 或 RxNE = 1 不产生任何中断 1： TxE = 1 或 RxNE = 1 生成事件中断(不管 DMAEN 的状态如何)
7	ITEVTEN	RW	0	事件中断启用 0： 事件中断禁用 1： 事件中断启用 以下情况下生成中断： – SB = 1 (主) – ADDR = 1 (主/从) – ADD10= 1 (主) – STOPF = 1 (从) – BTF = 1 无 TxE 或 RxNE 事件 – 如果 ITBUFEN = 1 TxE 事件为 1 – 如果 ITBUFEN = 1 RxNE 事件为 1
6	ITERREN	RW	0	错误中断启用 0： 错误中断禁用 1： 错误中断启用 以下情况下产生中断： – BERR = 1 – ARLO = 1 – AF = 1 – OVR = 1

5：0	FREQ_DIV	RW	0	时钟分频器 标准模式 6`b000000： 10 kHz 6`b000001： 20 kHz ... 6`b001001： 100 kHz Fast-mode 6`b001010： 110 kHz 6`b001011： 120 kHz ... 6`b100111： 400 kHz 高速模式 6`b101000： 500 kHz 6`b101001： 600 kHz ... 6`b101110： 1.5 MHz

偏移地址： 13-10h

I2C 状态寄存器

位	字段名	属性	默认	字段描述
31：16	-	-	0	预留的
15	GENCALL	RO	0	通用呼叫地址(从模式) 0： 无通用呼叫 Call 1： 当 ENGC=1 时接收通用呼叫地址
14	TRA	RO	0	发送器/接收器 0：接收到数据字节 1： 发送数据字节
13	BUSY	RO	0	总线忙 0： 总线上无通讯 1： 总线上正在进行通信

12	MSL	RO	0	主/从 0： 从模式 1： 主模式
11	OVR	RC_W0	0	超限/欠载 0： 无超限/欠载 1： 超限或者欠载
10	AF	RC_W0	0	确认失败 0： 未发生确认失败 1： 确认失败
9	ARLO	RC_W0	0	仲裁丢失(主模式) 0： 未检测到仲裁丢失 1： 检测到仲裁丢失
8	BERR	RC_W0	0	总线错误 0：无错误的起始或停止条件 1：错误的起始或停止条件
7	-	-	0	预留的
6	TXE	RO	0	数据寄存器空(发送器) 0： 数据寄存器非空 1： 数据寄存器空
5	RXNE	RO	0	数据寄存器非空(接收器) 0： 数据寄存器空 1： 数据寄存器非空
4	STOPF	RO	0	停止检测(从模式) 0： 未检测到停止条件 1： 检测到停止条件
3	ADD10	RO	0	已发送位头 (主模式) 0： 未发生 ADD10 事件 1： 主设备已发送第一个地址字节(头)

2	BTF	RO	0	字节传输完成 0： 数据字节传输未完成 1： 数据字节传输成功
1	ADDR	RO	0	发送(主模式)/匹配的地址 (从模式) 地址匹配(从) 0： 地址不匹配或未收到 1： 收到匹配的地址 发送地址(主) 0： 地址传输没有结束 1： 址传输结束
0	SB	RO	0	起始位(主模式) 0： 无起始条件 1：生成起始条件

偏移地址： 17-14h

I2C 调试数据寄存器

位	字段名	属性	默认	字段描述
31：22	APB_IF_DBG	RO	0	APB 接口调试数据
21：11	I2CS_DBG	RO	0	I2C 从调试数据
10：0	I2CM_DBG	RO	0	I2C 主调试数据

13 实时时钟(RTC)

13.1 概述

实时时钟(RTC)提供自动唤醒以管理低功耗模式。RTC 是一个独立的 BCD 定时器/计数器。它的主要功能是记录当前时间。RTC 提供带有可编程报警中断的时间时钟/日历。

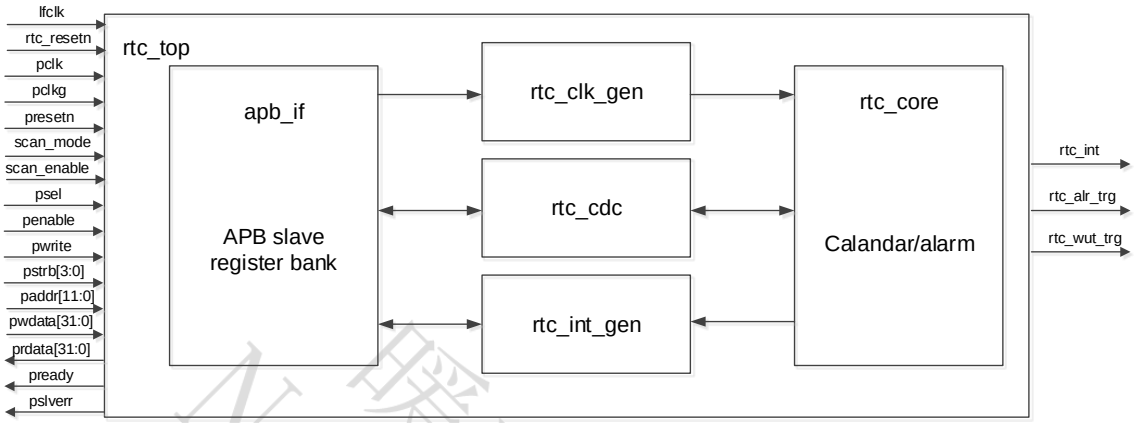
13.1.1 功能列表

- 计数秒、分、小时、日(星期几)、日期(每月中的第几天)、月和年
- 对 28 天、29 天(闰年)、30 天和 31 天进行补偿
- 二进制或 BCD 表示时间,日历和闹钟
- 12 小时或 24 小时时钟，上午和下午为 12 小时模式
- 两个中断是单独的软件掩码
  - 带可编程报警中断的时间时钟/日历
  - 周期性可编程定时器唤醒中断

13.2 框图

RTC 的功能框图 如下。

图 21 RTC 框图



- Rtc_apb_if 作为 apb 从模块负责生成寄存器组。
- rtc_clk_gen 基于 presscaler 生成 1Hz rtc 时钟输出 rtc_core。
- Rtc_cdc 处理 PCLK 和 RTC 时钟之间的时钟域交叉。
- Rtc_int_gen 生成 RTC 中断状态，包括闹钟和周期性唤醒。
- Rtc_core 是计算时间/日历的主要模块，包括世纪、年、月、日、周、时、分、秒。当编程时间/日期寄存器时，可以更新每个计数器。

13.3 函数描述

13.3.1 时钟和预分频器

在 RC 振荡器时钟和晶体振荡器时钟之间通过时钟控制器选择 RTC 时钟源(lfclk)。

一个可编程的预分频器位于偏移 0x1C 级，产生一个 1Hz 的时钟，用于更新日历。

分割后的 RTC 时钟由以下公式给出：

$$F_{rtc\_clk} = \frac{F_{lfclk}}{(prescaler + 1)}$$

当 lfclk 频率为 32.768KHz 时，默认预分频设置为 0x7FFF,那么分割后的 RTC 时钟是 1Hz。

13.3.2 实时时钟和日历

通过与 pclk (APB 时钟)同步的 RTC_TR 和 RTC_DR 寄存器访问 RTC 日历时间和日期寄存器。 RTC_TR 为时间，RTC_DR 为日期。

每个 RTC 时钟，当前日历值都会复制到 RTC_TR 和 RTC_DR 寄存器中。在停止或备用模式下不执行拷贝。

在初始模式下，当前日历将不会更新到 RTC_TR 和 RTC_DR 寄存器。退出初始模式后，当前日历将被更新到 RTC_TR 和 RTC_DR 寄存器。

13.3.3 时间/日历/闹钟数据模式

可以从 RTC 地址偏移 0x00 到 0x0D 访问时间、日历和闹钟寄存器。时间、日历和闹钟字节的内容可以是二进制或二进制编码的十进制(BCD)格式。在写入内部时间、日历、闹钟寄存器之前，先将控制寄存器的 INIT 位写入逻辑位，更新完成后再将 INIT 位写入逻辑零。初始化完成后，数据模式和小时模式不能再改变。当选择 12 小时格式时，小时字节的高阶位表示当它为逻辑格式时的 PM。

表 28 RTC 数据模式

地址位置	功能	十进制范围	范围	
			二进制数据模式	BCD 数据模式
0x00	秒	0-59	00-3B	00-59
0x01	分	0-59	00-3B	00-59
0x02	小时-12 小时模式	1-12	01-0C AM，81-8C PM	01-12AM，81-92PM
	小时-24 小时模式	0-23	00-17	00-23
0x03	星期几	1-7	01-07	01-07
0x04	本月几号	1-31	01-1F	01-31

0x05	月	1-12	01-0C	01-12
0x06	年	0-99	00-63	00-99
0x07	世纪	0-99	00-63	00-99
0x08	秒闹钟	0-59	00-3B	00-59
0x09	分闹钟	0-59	00-3B	00-59
0x0A	小时闹钟-12 小时	1-12	01-0C AM，81-8C PM	01-12AM，81-92PM
	小时闹钟-24 小时	0-23	00-17	00-23
0x0C	本月几号闹钟	1-31	01-1F	01-31
0x0D	月闹钟	1-12	01-0C	01-12

13.3.4 可编程闹钟

通过 RTC_CR 寄存器的 ALARM_EN 位开启可编程闹钟功能。当日历中的秒、分、时、日、月与闹钟寄存器 RTC_TAR、RTC_DAR 中的值相匹配时，RTC_ISR 中的 ALARM_STS 设置为 1。通过向该寄存器写入 1 来清除 ALARM_STS。

另一个使用条件是在一个或多个闹钟字节中插入“忽略”状态。对于秒、分、时闹钟，“忽略”代码是从 0xC0 到 0xFF 的任何十六进制值。对于月中的日闹钟，如果闹钟设置的范围不在 BCD 格式的 0x01-0x31 和二进制格式的 0x01-0x1F 之间，则视为“忽略”代码。对于月闹钟，如果闹钟设置的范围不在 BCD 格式的 0x01-0x12 和二进制格式的 0x01-0x0C 之间，则视为“忽略”代码。

通过 RTC_IER 寄存器中的 ALARM_IE 启用闹钟中断。

13.3.5 周期性唤醒

周期性唤醒标志由一个 16 位可编程自动重载向上计数器产生。周期性唤醒功能是通过 RTC_CR 寄存器中的 WUT_EN 位启用的。

可以通过在 RTC_CR 寄存器中配置 WUT_CLK_SEL 来选择唤醒定时器时钟输入：

- 当 WUT_CLK_SEL 为 0 时：通过配置 WUT_PRES，唤醒定时器时钟输入除以 1 ~ 16。当 LFCLK 为 32.768kHz 时，允许配置从 30us 到 32s 的唤醒中断周期。
- 当 WUT_CLK_SEL 为 1： 唤醒定时器时钟输入是 rtc_clk，它是 1Hz 的内部时钟，这允许在 1 秒到 36 小时之间实现唤醒，分辨率为 1 秒。

周期定时器的值通过写 RTC_PTR 寄存器来初始化，从 RTC_PTR 读取 RTC_PTR 时，该寄存器表示当前的周期定时器值。

如果当前周期定时器达到设置的值，则将 RTC_ISR 中的 WUT_STS 设置为 1。通过向该寄存器写入 1 来清除 WUT_STS。

通过 RTC_IER 寄存器中的 WUT_IE 启用周期性唤醒中断。

13.3.6 初始化与配置

日历初始化

要对初始时间和数据日历值进行编程，包括时间格式和预分配器配置，需要遵循以下顺序：

1) 要为日历计数器生成 1Hz 时钟，请编程 rtc 预分频器。

2) 将 INIT 位设置为 1，并在 RTC_CR 寄存器中配置 DATA_MODE 和 HOUR_MODE 以进入初始化模式。在这种模式下，日历计数器将停止，并可以更新其值。

3) 在 RTC_TR 和 RTC_DR 寄存器中加载初始时间和日期值。

4) 清除 INIT 位，退出初始化模式。在 2~3 个周期的 RTC 时钟之后(由于 pclk 到 rtc_clk 的同步)，加载实际的计数器值，然后日历开始计数。

5) 注意：当 RTC_SR 寄存器中的 INIT_SYNC_READY 位为零时，软件不能设置 INIT 位来触发另一次初始化。要在初始化后读取日历，软件必须检查是否设置了 INIT_SYNC_READY 位。

闹钟编程

必须遵循类似的程序来编程或更新可编程警报。

1) 编程时间闹钟 RTC_TAR 和日期警报 RTC_DAR 寄存器。

2) 在“RTC_CR 寄存器”中设置“ALARM_EN”为“启用告警”。如果需要产生中断，请在 RTC_IER 寄存器中设置 ALARM_IE。

周期性唤醒定时器编程

需要按照以下顺序配置或更改唤醒计时器值。

1) 在 RTC_CR 寄存器中编程定时唤醒输入时钟选择。

2) 在 RTC_WPR 寄存器中编程周期性唤醒时钟预分频器。

3) 对 RTC_WTR 寄存器中的周期性唤醒定时器值进行编程

4) 在 RTC_CR 寄存器中设置 WUT_EN 以启用唤醒定时器。

5) 如果需要产生中断，在 RTC_IER 寄存器中设置 WUT_IE。

13.4 寄存器

表 29 RTC 寄存器

偏移	首字母缩写	寄存器描述
00h	RTC_TR	RTC 时间寄存器
04h	RTC_DR	RTC 日期寄存器
08h	RTC_TAR	RTC 定时闹钟寄存器
0Ch	RTC_DAR	RTC 日期闹钟寄存器
10h	RTC_IER	RTC 中断启用寄存器
14h	RTC_ISR	RTC 中断状态寄存器
18h	RTC_CR	RTC 控制寄存器
1Ch	RTC_PR	RTC 预分频寄存器
20h	RTC_WTR	RTC 唤醒定时器寄存器
24h	RTC_WPR	RTC 唤醒预分频器寄存器
28h	RTC_SR	RTC 状态寄存器

偏移地址： 03-00h

RTC 时间寄存器

位	字段名	属性	默认	字段描述
31：24	REG_DAY_WEEK	RW	01h	写入时，将写入更新后的星期几 读取时，表示当前星期几
23：16	REG_HOUR	RW	0	写入时，将写入更新后的小时 读取时，表示当前小时
15：8	REG_MIN	RW	0	写入时，将写入更新后的分钟 读取时，表示当前分钟
7：0	REG_SEC	RW	0	写入时，将写入更新后的秒 读取时，表示当前秒

偏移地址： 07-04h

RTC 日期寄存器

位	字段名	属性	默认	字段描述
31：24	REG_CEN	RW	20h	写入时，将写入更新后的世纪 读取时，表示当前世纪
23：16	REG_YEAH	RW	21h	写入时，将写入更新后的年 读取时，表示当前年份
15：8	REG_MON	RW	01h	写入时，将写入更新后的月 读取时，表示当前月份
7：0	REG_DAY_MON	RW	01h	写入时，将写入更新后的该月的第几天 读取时，表示该月的第几天

偏移地址： 0B-08h

RTC 时间闹钟寄存器

位	字段名	属性	默认	字段描述
31：24	-	R0	0	预留的
23：16	REG_HOUR_ALM	RW	0	指示小时闹钟设置
15：8	REG_MIN_ALM	RW	0	指示分钟闹钟设置
7：0	REG_SEC_ALM	RW	0	指示秒闹钟设置

偏移地址： 0F-0Ch

RTC 日期闹钟寄存器

位	字段名	属性	默认	字段描述
31：16	-	RO	0	预留的
15：8	REG_MON_ALM	RW	01h	指示月闹钟设置
7：0	REG_DAY_ALM	RW	01h	指示该月的某一天闹钟设置

偏移地址： 13-10h

RTC 中断启用寄存器

位	字段名	属性	默认	字段描述
31：2	-	R0	0	预留的
1	WUT_IE	RW	0	启用周期唤醒定时器中断 0： 禁用 1： 启用
0	ALARM_IE	RW	0	启用闹钟中断 0： 禁用 1： 启用

偏移地址： 17-14h

RTC 中断状态寄存器

位	字段名	属性	默认	字段描述
31：2	-	RO	0	预留的
1	WUT_STS	RW1C	0	周期唤醒定时器中断状态，可通过对该位写入 1 来清除此位
0	ALARM_STS	RW1C	0	闹钟中断状态，该位可以通过写 1 来清除

偏移地址： 1B-18h

RTC 控制寄存器

位	字段名	属性	默认	字段描述
31：7	-	RO	0	预留的
5	WUT_CLK_SEL	RW	0	定时唤醒定时器输入时钟选择： 0： 将时钟除以 LFCLK 的 1 到 16 1： 1hz 内部时钟
4	WUT_EN	RW	0	启用定时唤醒定时器 0： 禁用 1： 启用

3	ALARM_EN	RW	0	启用闹钟 0： 禁用 1： 启用
2	DATA_MODE	RW	0	数据模式(DM)位表示时间和日历信息是二进制格式还是 BCD 格式。DM 位由程序设置为适当的格式，并可根据需要读取。DM 中的 1 表示二进制数据，而 DM 中的 0 表示二进制编码十进制(BCD)数据。
1	HOUR_MODE	RW	0	HOURS 24/12 模式 24/12 控制位建立小时字节的格式，1 表示 24 小时模式，0 表示 12 小时模式。
0	INIT	RW	0	当 INIT 位被写入 1 时，程序可以初始化时间和日历字节。初始化完成后，INIT 位被写入零。

偏移地址： 1F-1Ch

RTC 预分频器寄存器

位	字段名	属性	默认	字段描述
31：16	-	RO	0	预留的
15：0	PRESCALER	RW	7FFFh	RTC 时钟预分频器，所以当 LFCLK 为 32.768KHz 时，默认设置是将 RTC 时钟划分为 1Hz。 Frtc_clk = Flfclk/(PRESCALER+1)

偏移地址： 23-20h

RTC 唤醒定时器寄存器

位	字段名	属性	默认	字段描述
31：16	-	RO	0	预留的
15：0	WUT_VAL	RW	FFFFh	写入时，将写入定时定时器的初始值，默认为 FFFFh 读取时，表示当前周期定时器 只有当非零值有效时，才能产生周期性唤醒中断

偏移地址： 27h to 24h

RTC 唤醒预分频寄存器

位	字段名	属性	默认	字段描述
31：4	-	RO	0	预留的
3：0	WUT_PRES	RW	0	周期性唤醒时钟分频器，除以 1 到 16 Fwut_clk = Flfclk/(WUT_PRES+1)

偏移地址： 2Bh 至 28h

RTC 状态寄存器

位	字段名	属性	默认	字段描述
31：4	-	RO	0	预留的
3	WUT_VAL_SYNC_READY	RO	1	唤醒定时器值同步就绪，当该位为 0 时，软件不能更新另一个 wut 值。
2	WUT_PRES_SYNC_READY	RO	1	唤醒定时器时钟预分频同步就绪，当该位为 0 时，软件无法更新另一个 wut 预分频器。
1	RTC_PRES_SYNC_READY	RO	1	RTC 时钟预分频同步就绪，当该位为 0 时，软件不能更新另一个预分频器。
0	INIT_SYNC_READY	RO	1	初始化就绪指示器，当此位为 0 时，软件无法发出另一个初始化，也无法读取日历。

14 看门狗定时器(WDT)

14.1 概述

看门狗定时器(WDT)是一种监控程序运行的系统功能定时器。它有助于从错误中恢复，如失控或死锁代码。WDT 被配置为预定义的超时时间，并且在启用时不断运行。如果在超时时间内没有清除 WDT，它将发出一个系统复位。有一个预警中断，表示看门狗即将超时。

14.1.1 功能列表

- 如果看门狗定时器在超时时间之前没有被清除，则会触发系统复位
- 生成预警中断
- 32-位自由计时器
- 可配置超时时间
- 软件控制锁，防止意外的写访问

14.2 流程图

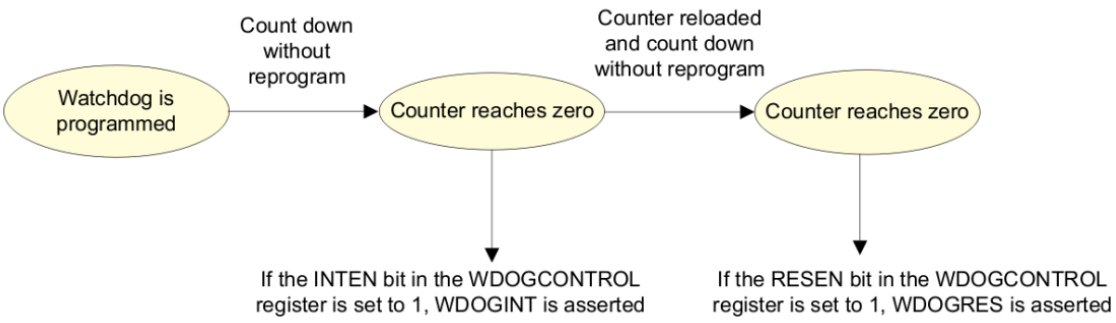
看门狗基于一个 32 位的向下数值计数器，该计数器由重载寄存器 WDOGLOAD 初始化。看门狗模块根据程序设定的值产生一个常规中断。在看门狗时钟的每一个上升沿上，计数器递减 1。

看门狗监视中断，如果在计数器达到 0 并且计数器停止之前中断没有被清除，则会发出一个复位请求信号。如果中断在达到零之前被清除，那么在下一个启用的看门狗时钟边缘上，计数器从 WDOGLOAD 寄存器重新加载，并继续计数。

看门狗模块在软件失效时对系统进行复位，提供了一种从软件崩溃中恢复的方法。

下图显示了看门狗操作的流程图。

图 22 WDT 流程图



14.3 函数描述

14.3.1 时钟配置

MCU 自由运行的 pclk(hclk 的分频器)是看门狗模块的输入时钟。通过在 SYSCTRL 寄存器中设置 WDT_CLKEN，可以启用 WDT 时钟。

14.3.2 操作

- WDOGLOCK 寄存器禁止对 WDT 寄存器的写入访问。这可以防止软件禁用看门狗功能。写入 0x1ACCE551 的值将启用对寄存器的写入访问。在写入 WDT 寄存器之前，通过将 0x1ACCE551 写入 WDOGLOCK 寄存器来解锁寄存器。
- WDT 定时器为每个 pclk 递减。使用下面的公式计算所需时间间隔(us)的计数器值：

  WDOGLOAD =间隔(us) * pclk (MHz)

  通过写入 WDOGLOAD[31：0]位来加载间隔。
- 加载的计数器值被重新加载到当前计数器(WDOGVALUE)寄存器上。向 WDOGCONTROL 寄存器的 INTEN 位写入“1”以启用看门狗和预警中断。当 WDT 计数器达到零时，触发中断。通过设置 WDOGMIS 位来指示中断状态。如果 WDOGINTCLR 寄存器被写入任何值，比如 ISR 例程中的“1”，那么 WDOGMIS 位将被清除。如果 RESEN 位为“0”，看门狗定时器重新加载 WDOGLOAD 值，并继续递减。
- 完成所有配置后，通过将 0x1ACCE551 以外的任何值写入 WDOGLOCK 寄存器来锁定寄存器。
- 如果 WDT 当前计数器没有重新加载初始值，那么当它达到零时，WDT 将触发一个早期预警中断。因此，在触发中断之前，需要用相同或新的值重新加载 WDOGLOAD 寄存器。

14.4 寄存器

表 30 WDT 寄存器

偏移	首字母缩写	寄存器描述
00h	WDOGLOAD	看门狗加载寄存器
04h	WDOGVALUE	看门狗值寄存器
08h	WDOGCONTROL	看门狗控制寄存器
0Ch	WDOGINTCLR	看门狗清除中断寄存器
10h	WDOGRIS	看门狗原始中断状态寄存器
14h	WDOGMIS	看门狗屏蔽中断状态寄存器
C00h	WDOGLOCK	看门狗锁寄存器
F00h	WDOGITCR	看门狗集成测试控制寄存器
F04h	WDOGITOP	看门狗集成测试输出集寄存器

偏移地址： 003-000h

看门狗加载寄存器

位	字段名	属性	默认	字段描述
31：0	WDOGLOAD	RW	FFFFFFFFh	包含计数器要递减的值，在写入寄存器时，立即从新值重新开始计数值

偏移地址： 007-004h

看门狗值寄存器

位	字段名	属性	默认	字段描述
31：0	WDOGVALUE	RO	FFFFFFFFh	给出递减计数器的当前值

偏移地址： 00B-008h

看门狗控制寄存器

位	字段名	属性	默认	字段描述
31：2	-	RO	0	预留的
1	RESEN	RW	0	启用看门狗复位输出
0	INTEN	RW	0	启用中断事件

偏移地址： 00F-00Ch

看门狗清除中断寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	WDOGINTCLR	WO	0	向 WDOGINTCLR 寄存器写入任何值都会清除看门狗中断，并从 WDOGLOAD 中的值重新加载计数器

偏移地址： 013-010h

看门狗原始中断状态寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	WDOGRIS	RO	0	指示来自计数器的原始中断状态，当从控制寄存器中启用中断时，该值为 ANDed，以创建屏蔽中断

偏移地址： 017-014h

看门狗屏蔽中断状态寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	WDOGMIS	RO	0	从计数器指示屏蔽中断状态

偏移地址： C03-C00h

看门狗锁寄存器

位	字段名	属性	默认	字段描述
31：0	WDOGLOCK	RW	0	写入 0x1ACCE551 的值将启用对所有其他寄存器的写访问。 写入任何其他值都会禁用写访问。从该寄存器读取数据只返回底部位： 0：表示启用写访问，不锁定 1：表示禁止写访问，锁定

偏移地址： F03-F00h

看门狗集成测试控制寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	WDOGITCR	RW	0	当设置为 1 时，看门狗进入集成测试模式

偏移地址： F07-F04h

看门狗集成测试输出集寄存器

位	字段名	属性	默认	字段描述
31：2	-	RO	0	预留的
1	TESTINT	RW	0	在集成测试模式下，WDOGINT 上输出的值
0	TESTRESET	RW	0	在集成测试模式下在 WDOGRES 上输出的值

15 脉冲宽度调制 (PWM)

15.1 概述

PWM 基于标准的定时器块，继承了它的所有特性，只有 PWM 功能够通过 PIN 脚输出信号。 该定时器设计用于对外设时钟(PCLK)的周期进行计数，并基于 7 个匹配寄存器，在发生指定的定时器值时可选地产生中断或执行其他操作。PWM 功能是对这些功能的补充，并且基于匹配寄存器事件。

单独控制上升沿和下降沿位置能够保证 PWM 用于更多的应用。 例如，多相电机控制通常需要三个不重叠的 PWM 输出，分别控制所有三个脉冲宽度和位置。

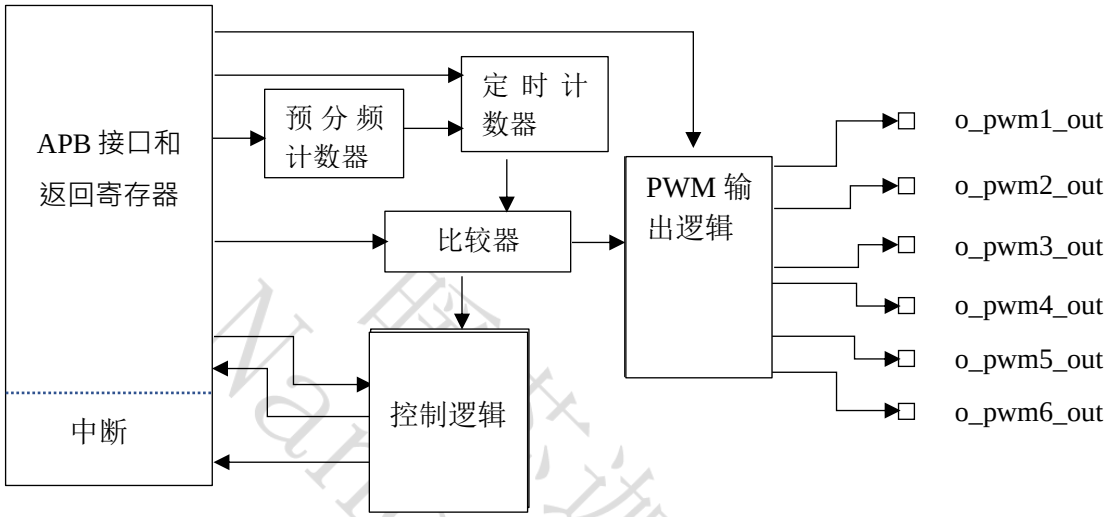
15.1.1 功能列表

- 七个匹配寄存器允许多达 6 个单边控制或 3 个双边控制 PWM 输出，或两种类型的混合。
- 支持单边控制和/或双边控制 PWM 输出。
- 脉冲周期和宽度可以是任意数量的计时器计数。
- 双边缘控制的 PWM 输出可以被编程为正向或负向脉冲。
- 匹配寄存器更新与脉冲输出同步，以防止产生错误的脉冲。
- 如果 PWM 模式未启用，可以用作标准定时器。
- 一个 32 位定时器/带可编程的 32 位预分频器的计数器。

15.2 框图

PWM 功能框图如下。

图 23 PWM 框图



15.3 函数描述

两个匹配寄存器可用于提供单边缘控制的 PWM 输出。一个匹配寄存器(PWMMR0)通过在匹配时重置计数来控制 PWM 周期速率。 另一个匹配寄存器控制 PWM 边缘位置。额外的单边缘控制 PWM 输出每个只需要一个匹配寄存器，因为所有 PWM 输出的重复率是相同的。当 PWMMR0 匹配发生时，多个单边控制的 PWM 输出在每个 PWM 周期开始时都有上升边。

三个匹配寄存器可以用来提供两个边缘控制的 PWM 输出。同样，PWMMR0 匹配寄存器控制 PWM 周期速率。另一个匹配寄存器控制两个 PWM 边缘位置。额外的双边缘控制 PWM 输出只需要每个匹配寄存器两个，因为所有 PWM 输出的重复率是相同的。

双边控制 PWM 输出，特定的匹配寄存器控制输出的上升和下降边缘。这样既可以产生正的 PWM 脉冲(上升沿先于下降沿出现)，也可以产生负的 PWM 脉冲(下降沿先于上升沿出现)。

15.3.1 单边缘控制 PWM 输出

在相应的 PWM 通道启用之后，除非它们的阴影匹配值等于 0，单边控制的 PWM 输出在 PWM 周期开始时变高电平。

当达到匹配值时，每个 PWM 输出会变低。如果没有匹配(即匹配值大于 PWM 速率)，则 PWM 输出持续保持高电平。

15.3.2 双边缘控制 PWM 输出

下一个 PWM 周期的匹配值在一个 PWM 周期结束时使用(与下一个 PWM 周期开始时一致的时间点)，规则 3 中指出的除外。

匹配值等于 0 或当前 PWM 速率(与匹配通道 0 值相同)具有相同的效果，规则 3 中注明的除外。例如，在 PWM 周期开始时要求下降沿与在 PWM 周期结束时要求下降沿具有相同的效果。

当匹配值发生变化时，如果“旧”匹配值中的一个等于 PWM 速率，如果新匹配值中的任何一个都不等于 0 或 PWM 速率，并且没有旧匹配值等于 0，则再次使用一次。

如果同时要求 PWM 输出的 set 设置和 clear 清除，则 clear 清除优先。当 set 和 clear 匹配值与 in 相同，或 set 或 clear 值等于 0 且另一个值等于 PWM 速率时，可能发生这种情况。

如果匹配值超出范围(即大于 PWM 速率值)，则没有匹配事件发生，匹配通道对输出没有影响。这意味着 PWM 输出将始终保持在一个状态，允许始终低电平，始终高电平，或“无变化”输出。

15.4 寄存器

外设寄存器必须通过字节(8 位)、半字(16 位)或字(32 位)访问。这些寄存器在设备内存映射中的 32 位地址可用。有关这些寄存器的内存地址，请参阅特定于设备的数据手册。

表 31 PWM 寄存器

偏移	首字母缩写	寄存器描述
00h	PWM_TC	PWM 定时器计数器寄存器
04h	PWM_TCR	PWM 定时器控制寄存器
08h	PWM_IR	PWM 中断寄存器
0Ch	PWM_PR	PWM 预分频寄存器
10h	PWM_PC	PWM 预分频计数器寄存器
14h	PWM_MCR	PWM 匹配控制寄存器
18h	PWM_MR0	PWM 匹配寄存器 0
1Ch	PWM_MR1	PWM 匹配寄存器 1
20h	PWM_MR2	PWM 匹配寄存器 2
24h	PWM_MR3	PWM 匹配寄存器 3
28h	PWM_MR4	PWM 匹配寄存器 4
2Ch	PWM_MR5	PWM 匹配寄存器 5
30h	PWM_MR6	PWM 匹配寄存器 6
34h	PWM_PCR	PWM 控制寄存器
38h	PWM_LER	PWM 加载启用寄存器
3Ch	PWM_DBG	PWM 调试寄存器

偏移地址： 03-00h

PWM 定时器寄存器

位	字段名	属性	默认	字段描述
31：0	PWM_TC	RW	0	定时器计数器

偏移地址： 07-04h

PWM 定时器寄存器

位	字段名	属性	默认	字段描述
31：2	-	-	0	预留的
1	CNT_EN	RW	0	启用计数器 0： 禁用计数器 1： PWM 定时计数和 PWM 预分频计数器
0	CNT_RET	RW	0	启用计数器复位 0： 清除复位 1： PWM 定时计数和 PWM 预分频计数器同步复位

偏移地址： 0B-08h

PWM 中断寄存器

位	字段名	属性	默认	字段描述
31：7	-	-	0	预留的
6	PWMMR6_INT	RW1C	0	PWM 匹配的中断标志 6
5	PWMMR5_INT	RW1C	0	PWM 匹配的中断标志 5
4	PWMMR4_INT	RW1C	0	PWM 匹配的中断标志 4
3	PWMMR3_INT	RW1C	0	PWM 匹配的中断标志 3
2	PWMMR2_INT	RW1C	0	PWM 匹配的中断标志 2
1	PWMMR1_INT	RW1C	0	PWM 匹配的中断标志 1
0	PWMMR0_INT	RW1C	0	PWM 匹配的中断标志 0

偏移地址： 0F-0Ch

PWM 预分频器寄存器

位	字段名	属性	默认	字段描述
31：0	PWM_PR	RW	0	预分频器寄存器

偏移地址： 13-10h

PWM 预分频器计数器

位	字段名	属性	默认	字段描述
31：0	PWM_PC	RW	0	预分频器计数器

偏移地址： 17-14h

PWM 匹配控制寄存器

位	字段名	属性	默认	字段描述
31：21	-	-	0	预留的
20	PWMMR6_STP	RW	0	0：禁用本功能 1：在 PWMMR6 上停止：如果 PWMMR6 与 PWMTC 匹配，则 PWMTC 和 PWMPC 将停止，PWMTCR[2]将设置为 0
19	PWMMR6_RET	RW	0	0：禁用本功能 1：对 PWMMR6 复位：如果 PWMMR6 匹配，则复位 PWMTC
18	PWMMR6_INT	RW	0	0： 禁用本中断 1： PWMMR6 上的中断：当 PWMMR6 与 PWMTC 中的值匹配时，产生中断
17	PWMMR5_STP	RW	0	0：禁用本功能 1：在 PWMMR5 上停止：如果 PWMMR5 与 PWMTC 匹配，则 PWMTC 和 PWMPC 将被停止，PWMTCR[2]将被设置为 0
16	PWMMR5_RET	RW	0	0：禁用本功能 1：在 PWMMR5 上复位：如果 PWMMR5 匹配，则复位 PWMTC
15	PWMMR5_INT	RW	0	0： 禁用本中断 1：PWMMR5 上的中断：当 PWMMR5 匹配 PWMTC 上的值时，产生中断
14	PWMMR4_STP	RW	0	0：禁用本功能 1：在 PWMMR4 上停止：如果 PWMMR4 与 PWMTC 匹配，则 PWMTC 和 PWMPC 将被停止，PWMTCR[2]将被设置为 0

13	PWMMR4_RET	RW	0	0：禁用本功能 1：在 PWMMR4 上复位：如果 PWMMR4 与 PWMMR4 匹配，则复位 PWMTC
12	PWMMR4_INT	RW	0	0： 禁用本中断 1：在 PWMMR4 上的中断：当 PWMMR4 匹配到 PWMTC 中的值时，产生中断
11	PWMMR3_STP	RW	0	0：禁用本功能 1：在 PWMMR3 上停止：如果 PWMMR3 与 PWMTC 匹配，则 PWMTC 和 PWMPC 将被停止，PWMTCR[2] 将被设置为 0
10	PWMMR3_RET	RW	0	0：禁用本功能 1：在 PWMMR3 上复位：如果 PWMMR3 匹配，则复位 PWMTC
9	PWMMR3_INT	RW	0	0： 禁用本中断 1： 在 PWMMR3 上的中断： 当 PWMMR3 匹配 PWMTC 中的值时，产生中断
8	PWMMR2_STP	RW	0	0：禁用本功能 1：在 PWMMR2 上停止：如果 PWMMR2 与 PWMTC 匹配，则 PWMTC 和 PWMPC 将停止，PWMTCR[2]将设置为 0
7	PWMMR2_RET	RW	0	0：禁用本功能 1：在 PWMMR2 上复位：如果 PWMMR2 与 PWMMR2 匹配，则复位 PWMTC
6	PWMMR2_INT	RW	0	0： 禁用本中断 1：在 PWMMR2 上的中断：当 PWMMR2 匹配 PWMTC 中的值时，产生中断
5	PWMMR1_STP	RW	0	0：禁用本功能 1：在 PWMMR1 上停止：如果 PWMMR1 与 PWMTC 匹配，则 PWMTC 和 PWMPC 将被停止，PWMTCR[2] 将被设置为 0
4	PWMMR1_RET	RW	0	0：禁用本功能 1：在 PWMMR1 上复位：匹配 PWMMR1 时，复位 PWMTC
3	PWMMR1_INT	RW	0	0： 禁用本中断 1： 在 PWMMR1 上的中断： 当 PWMMR1 匹配 PWMTC 中的值时，产生中断
2	PWMMR0_STP	RW	0	0：禁用本功能 1：在 PWMMR0 上停止：如果 PWMMR0 与 PWMTC 匹配，PWMTC 和 PWMPC 将停止，并且 PWMTCR[2] 将被设置为 0

1	PWMMR0_RET	RW	0	0：禁用本功能 1：在 PWMMR0 上复位：如果 PWMMR0 与 PWMMR0 匹配，则复位 PWMTC
0	PWMMR0_INT	RW	0	0： 禁用本中断 1：在 PWMMR0 上的中断：当 PWMMR0 与 PWMTC 中的值相匹配时，产生中断

偏移地址： 1B-18h

PWM 匹配寄存器 0

位	字段名	属性	默认	字段描述
31：0	PWM_MR0	RW	0	匹配寄存器 0

偏移地址： 1F-1Ch

PWM 匹配寄存器 1

位	字段名	属性	默认	字段描述
31：0	PWM_MR1	RW	0	匹配寄存器 1

偏移地址： 23-20h

PWM 匹配寄存器 2

位	字段名	属性	默认	字段描述
31：0	PWM_MR2	RW	0	匹配寄存器 2

偏移地址： 27-24h

PWM 匹配寄存器 3

位	字段名	属性	默认	字段描述
31：0	PWM_MR3	RW	0	匹配寄存器 3

偏移地址： 2B-28h

PWM 匹配寄存器 4

位	字段名	属性	默认	字段描述

31：0	PWM_MR4	RW	0	匹配寄存器 4

偏移地址： 2F-2Ch

PWM 匹配寄存器 5

位	字段名	属性	默认	字段描述
31：0	PWM_MR5	RW	0	匹配寄存器 5

偏移地址： 33-30h

PWM 匹配寄存器 6

位	字段名	属性	默认	字段描述
31：0	PWM_MR6	RW	0	匹配寄存器 6

偏移地址： 37-34h

PWM 控制寄存器

位	字段名	属性	默认	字段描述
31：11	-	-	0	预留的
10	PWMEN6	RW	0	0： PWM6 输出被禁用 1： PWM6 输出被启用
9	PWMEN5	RW	0	0： PWM5 输出被禁用 1： PWM5 输出被启用
8	PWMEN4	RW	0	0： PWM4 输出被禁用 1： PWM4 输出被启用
7	PWMEN3	RW	0	0： PWM3 输出被禁用 1： PWM3 输出被启用
6	PWMEN2	RW	0	0： PWM2 输出被禁用 1： PWM2 输出被启用

5	PWMEN1	RW	0	0： PWM1 输出被禁用 1： PWM1 输出被启用
4	PWMSEL6	RW	0	0： 为 PWM6 选择单边缘控制模式 1： 为 PWM6 输出选择双边缘控制模式
3	PWMSEL5	RW	0	0： 为 PWM5 选择单边缘控制模式 1：为 PWM5 输出选择双边缘控制模式
2	PWMSEL4	RW	0	0： 为 PWM4 选择单边缘控制模式 1： 为 PWM4 输出选择双边缘控制模式
1	PWMSEL3	RW	0	0： 为 PWM3 选择单边缘控制模式 1： 为 PWM3 输出选择双边缘控制模式
0	PWMSEL2	RW	0	0： 为 PWM2 选择单边缘控制模式 1：为 PWM2 输出选择双边缘控制模式

偏移地址： 3B-38h

PWM 加载启用寄存器

位	字段名	属性	默认	字段描述
31：7	-	-	0	预留的
6	PWMML6_EN	RW	0	0： 禁用加载 1： 写入到 PWM 匹配 6 寄存器的最后一个值将在定时器下一次由 PWM 匹配事件复位时生效
5	PWMML5_EN	RW	0	0： 禁用加载 1：写入 PWM 匹 5 寄存器的最后一个值将在定时器下一次由 PWM 匹事件复位时生效
4	PWMML4_EN	RW	0	0： 禁用加载 1： 写入 PWM 匹 4 寄存器的最后一个值将在定时器下一次由 PWM 匹事件复位时生效
3	PWMML3_EN	RW	0	0： 禁用加载 1： 写入 PWM 匹 3 寄存器的最后一个值将在定时器下一次由 PWM 匹事件复位时生效

2	PWMML2_EN	RW	0	0：禁用加载 1：写入 PWM 匹 2 寄存器的最后一个值将在定时器下一次由 PWM 匹事件复位时生效
1	PWMML1_EN	RW	0	0：禁用加载 1：写入 PWM 匹 1 寄存器的最后一个值将在定时器下一次由 PWM 匹事件复位时生效
0	PWMML0_EN	RW	0	0：禁用加载 1：写入 PWM 匹 0 寄存器的最后一个值将在定时器下一次由 PWM 匹事件复位时生效

偏移地址： 3F-3Ch

PWM 调试数据寄存器

位	字段名	属性	默认	字段描述
31：12	-	-	0	预留的
11：0	PWM_DBG	RO	0	调试数据

16 定时器

16.1 概述

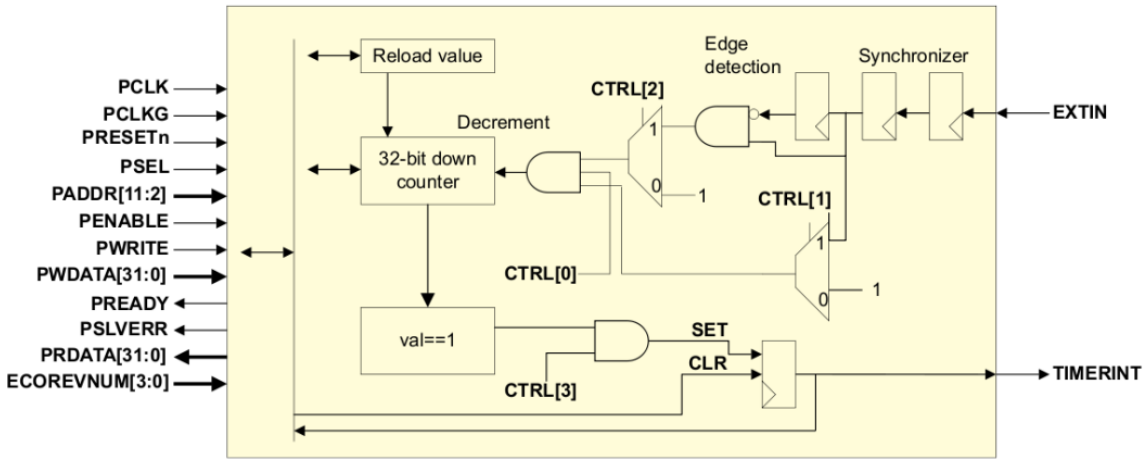
APB 定时器是一个 32 位下行计数器。

16.1.1 功能列表

- 当计数器到达 0 时，产生一个中断
- 当计数器达到 0 时重新加载计数器
- 支持外部引脚作为定时器启用
- 支持外部引脚作为时钟

16.2 框图

图 24 定时器框图



16.3 函数描述

当计数器达到 0 时，当设置了中断启用和计数器用重新加载寄存器 TMRRLD 更新时，生成一个中断。中断状态一直保持到通过向 TMRINT 寄存器写操作清除为止。写入 TMRVAL 寄存器时，更新当前值。

如果 APB 定时器数达到 0，同时软件清除之前的中断状态，则中断状态设置为 1。

在设置 EXT_EN 时，可以将外部输入信号 EXTIN 的从 0 到 1 的转换用作启用的定时器。

在设置 EXT_CLK 时，EXTIN 作为外部时钟输入，速度必须低于外设时钟的一半，因为它是通过一个双触发器进行采样，然后经过边缘检测逻辑。

16.4 寄存器

表 32 定时器寄存器

偏移	首字母缩写	寄存器描述
00h	TMRCTRL	定时器控制寄存器
04h	TMRVAL	定时器值寄存器
08h	TMRRLD	定时器重新加载寄存器
0Ch	TMRINT	定时器中断状态寄存器

偏移地址： 03-00h

定时器控制寄存器

位	字段名	属性	默认	字段描述
31：4	-	RO	0	预留的
3	INT_EN	RW	0	启用定时器中断
2	EXT_CLK	RW	0	选择外部输入为时钟
1	EXT_EN	RW	0	选择外部输入为启用
0	TMR_EN	RW	0	定时器启用

偏移地址： 07-04h

定时器值寄存器

位	字段名	属性	默认	字段描述
31：0	VALUE	RW	0	写入时，用该寄存器值更新当前计数器值 读取时，给定当前计数器值

偏移地址： 0B-08h

定时器重新加载寄存器

位	字段名	属性	默认	字段描述
31：0	RELOAD	RW	0	重新加载值，对该寄存器的写操作将在其达到 0 后设置当前值

偏移地址： 0F-0Ch

定时器中断状态寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	INT_STS	RW	0	定时器中断状态，写一个来清除

17 双定时器

17.1 概述

双定时器是一个 APB 双输入定时器模块，由两个可编程的 32 位或 16 位下位计数器(TIMER1，TIMER2)组成，当它们达到零时可以产生中断。每个定时器模块的操作是相同的。

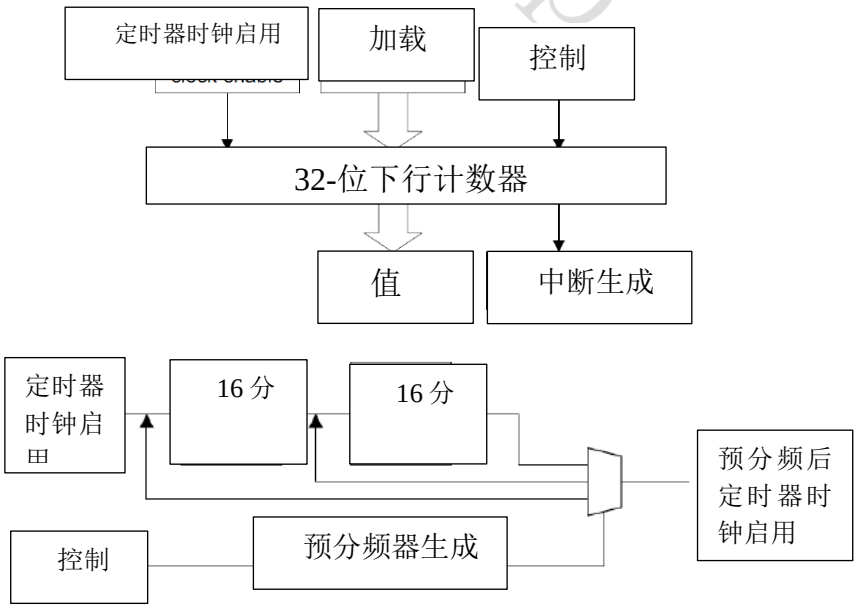
17.1.1 功能列表

- 可选配置
  - 16 或 32-位下行计数器操作
- 三种不同的操作模式
  - 自由运行模式
  - 单次计数模式
  - 周期模式
- 计数器下溢中断
- 内部三个可选预分频器

17.2 框图

以下为双定时器和预分频器时钟启用生成框图。

图 25 双定时器框图



17.3 函数描述

17.3.1 自由运行模式

计数器在达到零后自动填充，并在每次达到零时生成一个中断。它还从最大值开始继续倒数。这是默认模式。

17.3.2 周期定时器模式

计数器以固定的间隔产生一个中断，在超过 0 之后重新加载原始值。

17.3.3 单次定时器模式

计数器生成一次中断。当计数器达到 0 时，它将停止，直到重新编程。您可以使用以下方法之一来做到这一点：

清除控制寄存器中的单次计数位，在这种情况下，计数根据自由运行模式或周期模式的选择进行。

向加载值寄存器写入一个新值。

17.3.4 操作

每个定时器的操作都是相同的，并且有一组相同的寄存器。

- 通过配置 TIMER_SIZE 位来选择 16 位或 32 位计时器操作。
- 通过配置 ONE_SHOT，TIMER_MODE 位来选择定时器的操作模式。
- 通过向 INT_EN 位写入" 1 "来启用定时器中断。
- 计数器对每个 PCLK 减 1。使用以下公式计算所需间隔(ms)的计数器值：

  TIMn_LOAD = 间隔(ms) * PCLK (KHz)
- 通过写入 TIMn_LOAD 位来加载时间间隔。
- 加载的计数器值重新加载到 TIMn_VAL 寄存器的当前计数器上。
- 通过向 TIMER_EN 位写入“1”来启用定时器。
- 加载的计数器值向下计数为 0。
- 当达到 0 时，产生一个中断。这通过 TIMn_MIS 寄存器中被屏蔽的中断状态位表示。可以通过写入 TIMn_INTCLR 寄存器来清除中断。如果中断不可用，当计数器达到 0 时，原始中断状态寄存器 TIMn_RIS 表示计数器过期。
- 在单触发模式下，定时器到达 0 时停止。在周期模式下，计数器继续递减到 0，然后从 TIMn_LOAD 寄存器中重新加载 TIMn_VAL 并继续递减。在这种模式下，定时器有效地产生一个周期性中断。如果定时器处于自由运行模式，则从其最大值开始递减(32 位模式为 0xFFFFFFFF，16 位模式为 0xFFFF)。

17.4 寄存器

表 33 双定时器寄存器

偏移	首字母缩写	寄存器描述
00h	TIM1_LOAD	定时器 1 加载寄存器
04h	TIM1_VAL	定时器 1 当前值寄存器
08h	TIM1_CTRL	定时器 1 控制寄存器
0Ch	TIM1_INTCLR	定时器 1 中断清除寄存器
10h	TIM1_RIS	定时器 1 原始中断状态寄存器
14h	TIM1_MIS	定时器 1 屏蔽中断状态寄存器
18h	TIM1_BGLOAD	定时器 1 后台加载寄存器
20h	TIM2_LOAD	定时器 2 加载寄存器
24h	TIM2_VAL	定时器 2 当前值寄存器
28h	TIM2_CTRL	定时器 2 控制寄存器
2Ch	TIM2_INTCLR	定时器 2 中断清除寄存器
30h	TIM2_RIS	定时器 2 原始中断状态寄存器
34h	TIM2_MIS	定时器 2 屏蔽中断状态寄存器
38h	TIM2_BGLOAD	定时器 2 后台加载寄存器

偏移地址： 03-00h

定时器 1 加载寄存器

位	字段名	属性	默认	字段描述
31：0	TIM1_LOAD	RW	0	这个寄存器包含计数器要递减的值， 这是在启用周期模式且当前计数达到 0 时用于重新加载计数器的值。

偏移地址： 07-04h

定时器 1 当前值寄存器

位	字段名	属性	默认	字段描述
31：0	TIM1_VAL	RO	32’hFFFF_FFFF	该寄存器提供递增计数器的当前值

偏移地址： 0B-08h

定时器 1 控制寄存器

位	字段名	属性	默认	字段描述
31：8	-	RW	0	预留的
7	TIMER_EN	RW	0	启用位： 0： 禁用定时器 1： 启用定时器
6	TIMER_MODE	RW	0	模式位： 0： 定时器处于自由运行模式 1：定时器处于周期模式
5	INT_EN	RW	1	启用中断： 0： 禁用定时器中断 1： 启用 定时器中断
4	-	RO	0	预留的
3：2	TIMER_PRE	RW	00b	预分频器位： 00： 预分频 0 阶段，时钟 1 分频 01： 预分频 4 阶段，时钟 16 分频 10，11：预分频 8 阶段，,时钟 256 分频
1	TIMER_SIZE	RW	0	选择 16 位或 32 位计数器操作： 0： 16-位计数器 1： 32-位计数器
0	ONE_SHOT	RW	0	选择单次触发模式或者循环计数器模式： 0： 循环模式 1：单次触发模式

偏移地址： 0F-0Ch

定时器 1 中断清除寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	TIM1_INTCLR	WO	0	任何对该寄存器的写入都会清除计数器的中断输出

偏移地址： 13-10h

定时器 1 原始中断状态寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	TIM1_RIS	RO	0	该寄存器表示计数器的原始状态，当定时器控制寄存器启用定时器中断时，此值为 ANDed，以创建屏蔽中断，并将其传递给中断输出引脚。

偏移地址： 17-14h

定时器 1 屏蔽中断状态寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	TIM1_MIS	RO	0	该寄存器表示计数器的屏蔽状态，这个值是原始中断状态的逻辑与，从定时器控制寄存器启用了计时器中断，并且与传递给中断输出引脚的值相同。

偏移地址： 1B-18h

定时器 1 后台加载寄存器

位	字段名	属性	默认	字段描述
31：0	TIM1_BGLOAD	RW	0	这个寄存器包含计数器要递减的值，这是在启用周期模式且当前计数达到 0 时用于重新加载计数器的值。 该寄存器提供了一种访问 TIM1_LOAD 寄存器的替代方法，不同之处在于，写入 TIM1_BGLOAD 不会导致计数器立即从新值重新启动。从这个寄存器读取返回的值与从 TIM1_LOAD 返回的值相同。

偏移地址： 1F-1Ch

预留的

位	字段名	属性	默认	字段描述
31：0	-	RO	0	预留的

偏移地址： 23-20h

定时器 2 加载寄存器

位	字段名	属性	默认	字段描述
31：0	TIM1_LOAD	RW	0	这个寄存器包含计数器要递减的值，这是在启用周期模式且当前计数达到 0 时用于重新加载计数器的值。

偏移地址： 27-24h

定时器 2 当前值寄存器

位	字段名	属性	默认	字段描述
31：0	TIM2_VAL	RO	32’hFFFF_FFFF	这个寄存器提供正在递减的计数器的当前值

偏移地址： 2B-28h

定时器 2 控制寄存器

位	字段名	属性	默认	字段描述
31：8	-	RW	0	预留的
7	TIMER_EN	RW	0	启用位： 0： 禁用定时器 1：启用定时器
6	TIMER_MODE	RW	0	模式位： 0：处于自由运行模式的定时器 1： 处于周期模式的定时器

5	INT_EN	RW	1	启用中断： 0： 禁用定时器中断 1： 启用定时器中断
4	-	RO	0	预留的
3：2	TIMER_PRE	RW	00b	预分频器位： 00： 预分频 0 阶段，时钟 1 分频 01： 预分频 4 阶段，时钟 16 分频 10，11：预分频 8 阶段，时钟 256 分频
1	TIMER_SIZE	RW	0	选择 16 位或 32 位计数器操作： 0： 16-位计数器 1： 32-位计数器
0	ONE_SHOT	RW	0	选择单次触发模式或者循环计数器模式： 0： 循环模式 1：单次触发模式

偏移地址： 2F-2Ch

定时器 2 中断清除寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	TIM2_INTCLR	WO	0	对该寄存器的任何写入都将清除计数器的中断输出

偏移地址： 33-30h

定时器 2 原始中断状态寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	TIM2_RIS	RO	0	该寄存器表示计数器的原始状态。当定时器控制寄存器启用定时器中断时，此值为 ANDed，以创建屏蔽中断，并将其传递给中断输出引脚。

偏移地址： 37-34h

定时器 2 屏蔽中断状态寄存器

位	字段名	属性	默认	字段描述
31：1	-	RO	0	预留的
0	TIM2_MIS	RO	0	该寄存器表示计数器的屏蔽状态，这个值是原始中断状态的逻辑与，从定时器控制寄存器启用了计时器中断，并且与传递给中断输出引脚的值相同。

偏移地址： 3B-38h

定时器 2 后台加载寄存器

位	字段名	属性	默认	字段描述
31：0	TIM2_BGLOAD	RW	0	这个寄存器包含计数器要递减的值。这是在启用周期模式且当前计数达到 0 时用于重新加载计数器的值。 该寄存器提供了一种访问 TIM2_LOAD 寄存器的替代方法。不同之处在于，写入 TIM2_BGLOAD 不会导致计数器立即从新值重新启动。从这个寄存器读取返回的值与从 TIM2_LOAD 返回的值相同。

18 模拟控制 (ANAC)

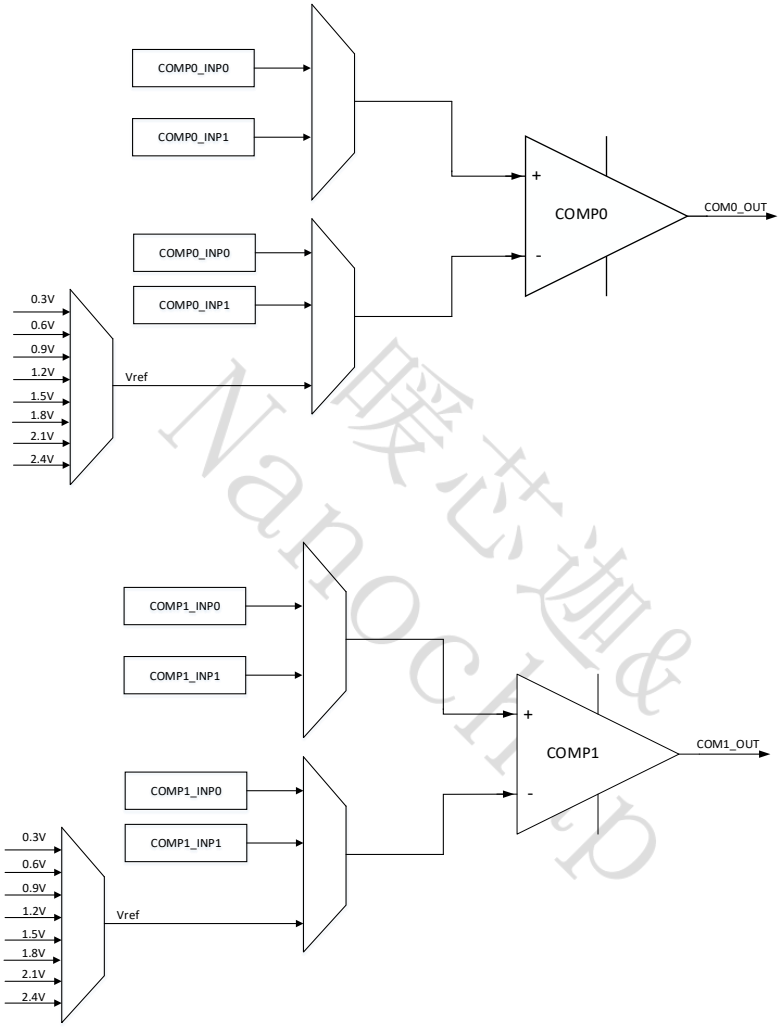
18.1 比较器

18.1.1 功能列表

- 轨到轨比较器
- 灵活的输入选择
  - 从 IO 口输入
  - 内部 Vref 输入
- 唤醒 MCU 的异步中断事件
- 比较器输出到 GPIO，用于指示比较器状态

18.1.2 框图

图 26 比较器框图



18.1.3 函数描述

比较器比较 Vin+和 Vin-，当 Vin+ > Vin-时，输出“1”，否则输出“0”。

Vin+可来自多 IO 输入，Vin-可来自多 IO 输入或内部参考电压。

比较器输出产生异步中断，无需时钟，可从低功耗模式唤醒 MCU。

表 34 COMP0 引脚和寄存器

COMP0	引脚	引脚选择寄存器	功能选择寄存器
COMP0_INP0	GPIO3	GPIO3_ALTF=2’b11 & GPIO_ANAE[3]=1’b1	
COMP0_INP1	GPIO22	GPIO22_ALTF=2’b11 & GPIO_ANAE[22]=1’b1	
COMP0_INN0	GPIO5	GPIO5_ALTF=2’b11 & GPIO_ANAE[5]=1’b1	
COMP0_INN1	GPIO23	GPIO23_ALTF=2’b11 & GPIO_ANAE[23]=1’b1	
COMP0_OUT	GPIO20	GPIO20_ALTF=2’b10	

表 35 COMP1 引脚和寄存器

COMP1	引脚	引脚选择寄存器	功能选择寄存器
COMP1_INP0	GPIO12	GPIO12_ALTF=2’b11 & GPIO_ANAE[12]=1’b1	
COMP1_INP1	GPIO14	GPIO14_ALTF=2’b11 & GPIO_ANAE[14]=1’b1	
COMP1_INN0	GPIO13	GPIO13_ALTF=2’b11 & GPIO_ANAE[13]=1’b1	
COMP1_INN1	GPIO19	GPIO19_ALTF=2’b11 & GPIO_ANAE[19]=1’b1	
COMP1_OUT	GPIO21	GPIO21_ALTF=2’b10	

18.2 寄存器

表 36 模拟控制寄存器

偏移	首字母缩写	寄存器描述
00h	COMP0_CTRL	比较器 0 控制寄存器
04h	COMP1_CTRL	比较 1 控制寄存器
08h	PGA_CTRL	PGA 控制寄存器
0Ch	CHARGE_CTRL	充电控制寄存器
10h	PMU_CTRL	PMU 控制寄存器
14h	BOOST_CTRL	引导控制寄存器
18h	ANA_BIST	模拟 BIST 选择寄存器

偏移地址： 03-00h

比较器 0 控制寄存器

位	字段名	属性	默认	字段描述
31：9	-	RO	0	预留的
8	COMP0_OUT	RO	0	比较器 0 输出
7	COMP_REF_EN	RW	0	比较器 0/1 参考电压启用 0： 禁用 1： 启用

6：4	COMP0_VREF_SEL	RW	0	比较器 0 参考电压选择 000： 0.3V 001： 0.6V 010： 0.9V 011： 1.2V 100： 1.5V 101： 1.8V 110： 2.1V 111： 2.4V
3：2	COMP0_SIGSEL_N	RW	0	比较器 0 负输入选择 00： COMP0_VIN0 01： COMP0_VIN1 10： Vref 11： 预留的
1	COMP0_SIGSEL_P	RW	0	比较器 0 正输入选择 0： COMP0_VIP0 1： COMP0_VIP1
0	COMP0_EN	RW	0	比较器 0 启用 0： 禁用 1： 启用

偏移地址： 07-04h

比较器 1 控制寄存器

位	字段名	属性	默认	字段描述
31：9	-	RW	0	预留的
8	COMP1_OUT	RO	0	比较器 1 输出
7	-	RO	0	

6：4	COMP1_VREF_SEL	RW	0	比较器 1 参考电压选择 000： 0.3V 001： 0.6V 010： 0.9V 011： 1.2V 100： 1.5V 101： 1.8V 110： 2.1V 111： 2.4V
3：2	COMP1_SIGSEL_N	RW	0	比较器 1 负输入选择 00： COMP1_VIN0 01： COMP1_VIN1 10： Vref 11： 预留的
1	COMP1_SIGSEL_P	RW	0	比较器 1 正输入选择 0： COMP1_VIP0 1： COMP1_VIP1
0	COMP1_EN	RW	0	比较器 1 启用 0：禁用 1： 启用

偏移地址： 0B-08h

PGA 控制寄存器

位	字段名	属性	默认	字段描述
31：14	-	RO	0	预留的

13：12	PGA_VIN_SEL	RW	0	PGA 负输入选择 00：PGA_VIN0 01：PGA_VIN1 10：内部 VCM 11：外部 VCM
11	-	RO	0	预留的
10：8	PGA_VIP_SEL	RW	0	PGA 正输入选择 000：PGA_VIP0 001：PGA_VIP1 010：内部 VCM 011： VBAT 100： AVDD1P8 101： VLCD[0] 110： VREF1P2V 111： V_temp
7	-	RO	0	预留的
6：4	PGA_GAIN_SEL	RW	0	PGA 增益选择位 在逆变模式下，增益从 1 到 8 在非逆变模式下，增益为 2 ~ 9
3：2	-	RO	0	
1	PGA_REF_EN	RW	0	PGA 参考启用 0： 禁用 1： 启用
0	PGA_OP_EN	RW	0	PGA 启用 0： 禁用 1： 启用

偏移地址： 0F-0Ch

充电控制寄存器

位	字段名	属性	默认	字段描述
31：11	-	RO	0	预留的
10：8	CHARGE_CURRENTSEL	RW	010b	设置充电电流值的信号 000： 133mA 001： 145mA 010： 158mA 011： 174mA 100： 193mA 101： 219mA 110： 250mA 111： 292mA
7	-	RO	0	预留的
6：4	CHARGE_LVSEL	RW	100b	设置低电压检测值信号： 000： 300mV 001： 600mV 010： 900mV 011： 1200mV 100： 1500mV 101： 1800mV 110： 2100mV 111： 2400mV
3	CHARGER_END	RO	0	当充电完毕，这个信号从“0”变为“1”。
2	CHARGER_OK	RO	0	当充电电源就绪时，信号由“0”变为“1”
1	CHARGE_BATON	RW	0	充电功能开启启用位 0： 禁用 1： 启用

0	CHARGE_SYSON	RW	0	开启充电器 LDO 的启用位 0： 禁用 1： 启用

偏移地址： 13-10h

PMU 控制寄存器

位	字段名	属性	默认	字段描述
31：8	-	RO	0	预留的
7	TEMP_150C_TRIG	RO	0	当芯片温度为 150C 时，信号从 0 变为 1
6	LVD_OUT	RO	0	当电源低于阈值时，信号从 0 变为 1
5	TEMP_EN	RW	0	温度传感器开启启用位 0： 禁用 1： 启用
4	BG_BUFFER_EN	RW	0	启用能隙缓冲区的位 0： 禁用 1： 启用
3：1	LVD_INSEL	RW	0	为 LVD 选择不同电压的信号 000： 4.2V 001： 3.9V 010： 3.6V 011： 3.3V 100： 3.0V 101： 2.7V 110： 2.4V 111： 2.1V
0	LVD_EN	RW	0	开启低压检测器的启用位 0： 禁用 1： 启用

偏移地址： 17-14h

预留的

偏移地址： 1B-18h

模拟 BIST 选择寄存器

位	字段名	属性	默认	字段描述
31：16	-	RO	0	预留的
15：8	ANA_RSV	RO	0	模拟预留信号
7	-	RO	0	-
6：4	BIST_SEL	RW	0	选择不同的信号到模拟 bist 输出 000： 能隙电流 001： VBG 010： 比较器为 1.2V 011： VCM (vdd/2)用于 pga 100： vcm (vdd/2)用于 adc 101： 直流-直流升压装置的温度电压 110： 预留的 111： 预留的
3：1	-	RO	0	
0	BIST_EN	RW	0	为模拟 bist 启用位 0： 禁用 1： 启用

19 液晶显示控制器（LCD）

19.1 概述

LCD 控制器是单色无源液晶显示器(LCD)的数字控制器/驱动器，具有多达 4 个公共终端和多达 16 个分段终端。

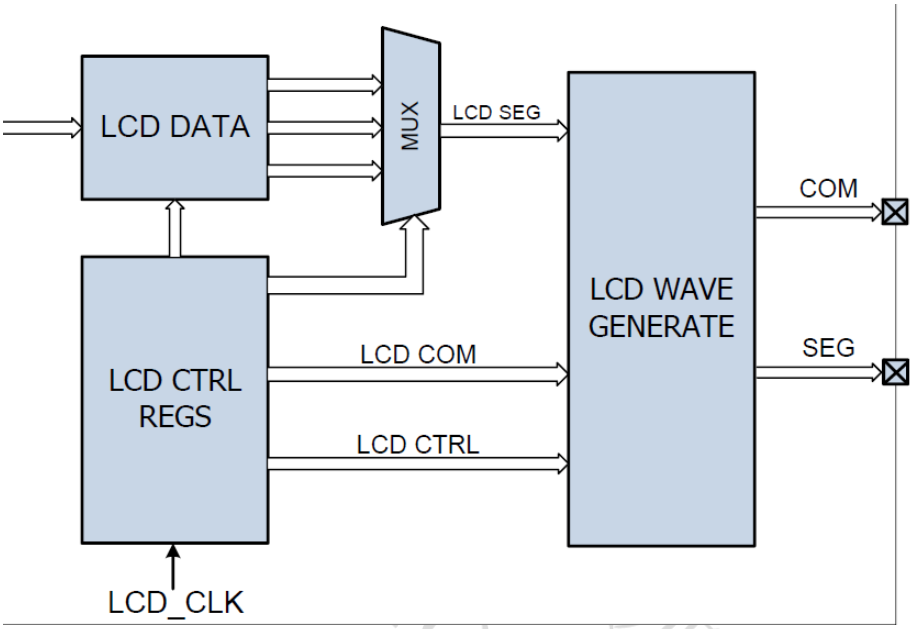
LCD 总共需要 20 个 I/O，包括 4 个 COM 和 16 个 SEG，通过 GPIO 的复用实现。在启用 LCD 之前，将相关的 GPIO_ALTF[1：0]设置为 2 'b11，则 GPIO 用于模拟功能。然后启用 LCD 和相关的 COMEN 和 SEGEN。

19.1.1 功能列表

- 支持多达 4 个公共终端和 16 个分段终端
- 支持 1/3 偏压
- 支持 1/4 占空
- 支持 16 灰度
- 支持闪烁，闪烁频率可配置
- 支持间歇性亮灯
- 支持所有屏幕亮起，所有屏幕熄灭
- LCD 驱动器可在激活模式、睡眠模式和停止模式下工作
- 支持类型 A 驱动波
- 典型刷新频率为 64Hz

19.2 框图

图 27 LCD 框图



20 电刺激模块

20.1 刺激参数

20.1.1 刺激通道

项目	值	单位
电源电压	5~60	V
电荷不平衡	<10	uC/sec
	<0.75uA	mm2
电流阵列	8 个带有电流和开关的驱动器(8 个电流源，8 个电流汇入)	
电流范围	0~67，8 位	mA
刺激频率	1~250K	Hz
脉冲宽度	2~infinity	us
单位电流	33~264，33steps	uA

20.1.2 波形示例

图 28 方波形

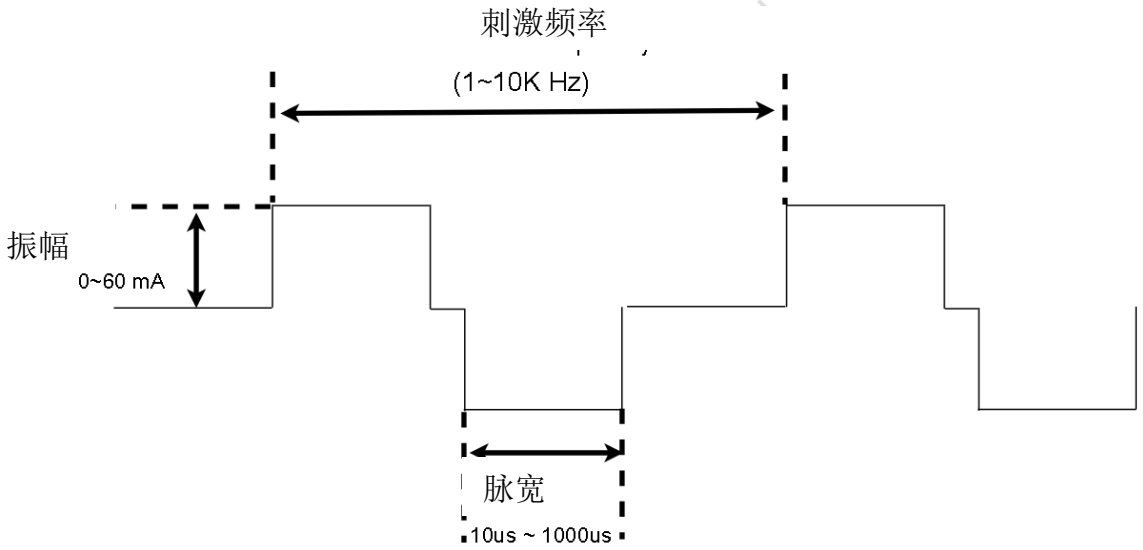


图 29 正弦波形

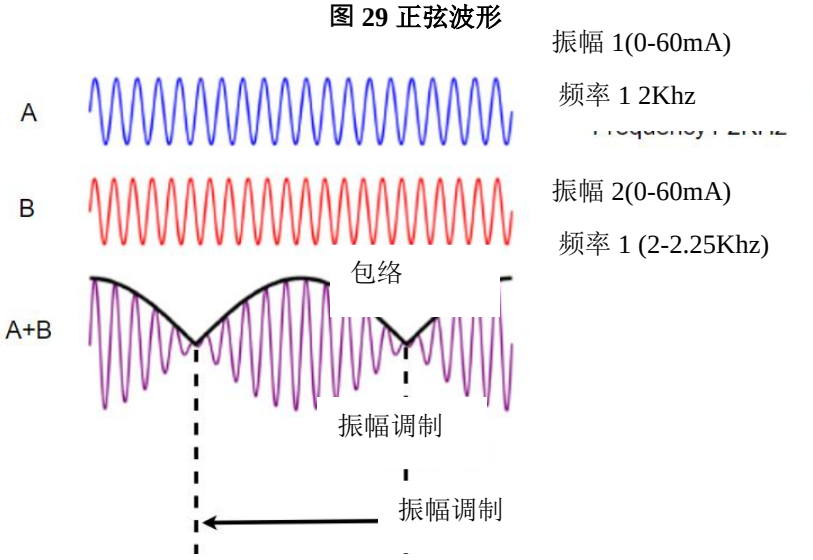
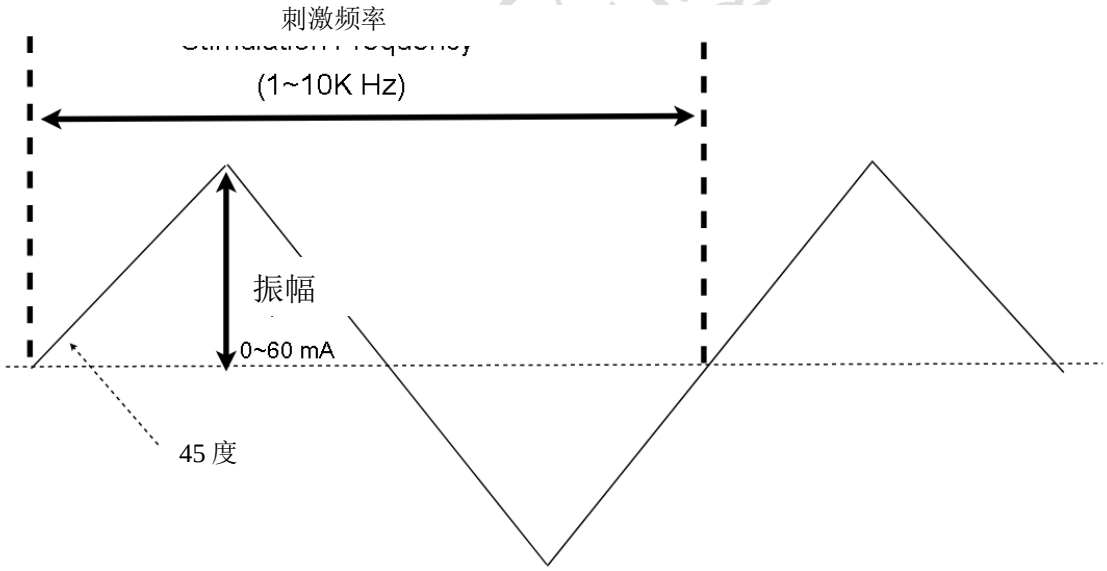


图 30 三角波形



20.1.3 波形寄存器

名称	位范围	访问	定义
ADDR_WG_DRV_CONFIG_REG	<7：0>	RD/WR	位 0：静息启用，1： 负启用，2： 静音启用，3：源 B 启用，4： 交替(+/-)正面， 5： 如果一个与中断一起工作，如果是零，继续重复波形，6：多电极
ADDR_WG_DRV_CTRL_REG	<7：0>	RD/WR	仅 0 位：启用波生成块；其余未使用
ADDR_WG_DRV_REST_T_REG	<7：0>	WR	在一个周期内，波的正侧和负侧之间的静息时间(单位为微秒)
ADDR_WG_DRV_SILENT_T_REG	<31：0>	WR	下一波周期前的静默时间(微秒)
ADDR_WG_DRV_HLF_WAVE_PRD_REG	<31：0>	WR	任意波(如正弦波或方波)周期的一半(以微秒为单位)
ADDR_WG_DRV_NEG_HLF_WAVE_PRD_REG	<31：0>	WR	任意波(如正弦波或方波)周期的负一半(以微秒为单位)
ADDR_WG_DRV_CLK_FREQ_REG	<7：0>	WR	时钟的频率，单位为 MHz
ADDR_WG_DRV_IN_WAVE_ADDR_REG	<7：0>	WR	将波形值的下一个 8 位写入存储波形值的 APB 寄存器文件的地址
ADDR_WG_DRV_IN_WAVE_REG	<7：0> <15：8>	WR	下一个半波点的值写入 ADDR_WG_DRV_IN_WAVE_ADDR_REG 指定的地址，总半波是 64 个点，每个点 8 位在 0 到 255 之间，如果使用驱动器 B，用户可以选择激活驱动器 B 的哪个通道，同时发送半波点<7：0>。也就是说，对于半波的每个值，可以选择唯一的通道(在 0 到 23 之间)。该值(通道号)以<15：8>位保存(仅适用于驱动程序 B)。
ADDR_WG_DRV_ALT_LIM_REG	<15：0>	WR	一个周期内交替信号的时钟数
ADDR_WG_DRV_ALT_SILENT_LIM_REG	<15：0>	WR	每个沉默持续时间的时钟数为交替频率
ADDR_WG_DRV_DELAY_LIM_REG	<15：0>	WR	禁用复位后和产生波之前用于初始延迟的时钟数。这种延迟适用于有多个驱动器的情况，如 A 和 C，因此我们可以在它们开始之前对每个驱动器设置延迟，以防止它们重叠。 然而，在驱动程序 b 中，只有一个驱动程序(有 24 个通道)。拖延只会推迟

			开始，这是没有用的，因为我们会推迟助听器声波产生的开始，而它应该尽快开始。
ADDR_WG_DRV_NEG_SCALE_REG	<7：0>	WR	用这个无符号值缩放波形的负侧(乘以这个值)
ADDR_WG_DRV_NEG_OFFSET_REG	<7：0>	WR	用这个无符号值偏移(移位)波形的负侧
ADDR_WG_DRV_INT_REG	<31：0>	RD/WR	写入访问： 位 0：启用中断过程 位 1： 启用时清除第一个地址中断 位 2： 当启用时，清除第二地址中断 位 <15：8>： 第一个地址中断，当波形到达该波形地址时，启用 APB 中断信号(有 64 个点波形，因此 64 个地址用作第一地址中断)。 位 <23：16>： 第二个地址中断。当波源到达该波形地址时，启用 APB 中断信号(有 64 个点波形，因此有 64 个地址用作第二地址中断)，剩余位保留。 读取访问： 位 <7：0>： 我们正在读取波发生器的数 位 <8>：中断启用 位 <9>： 发生第一个地址中断，当波形生成到达第一个波形地址时启用 位 <10>： 发生第二个地址中断，当波生成到达第二个波形地址时启用 位 <23：16>： 报告中断的第一个地址 位 <31：24>： 报告中断的第二个地址
ADDR_WG_DRV_ISEL_REG	<2：0>	WR	当前选择值
ADDR_WG_DRV_SW_CONFIG_REG	<7：0>	WR	电极开关的选择。该电极应使用哪个开关。 位 0： 用于电极的开关 0。位 1： 用于电极等的开关 1。 可以使用位的组合。

21 联系方式

可通过以下方式了解更多产品详情：

1) 公司电话：4008605922； 180 9470 6680

2) 技术人员 QQ：1708154204

3) 公众号：暖芯迦电子